Tecnologías de la información

Victor Cavaller Reyes (coordinador)
Lluís Anaya Torres
Miquel Barceló García
Ramon Costa i Pujol
Jaume Raventós Moret
Xavier Sánchez Porras
Ignasi Sebastià Oriol
Jeroni Vivó i Plans

PID_00153111

Material docente de la UOC

Universitat Oberta de Catalunya
www.uoc.edu

Victor Cavaller Reyes

Doctor por la Universidad de Barcelona dentro del Programa "Información y documentación en la era digital" del Departamento de Biblioteconomía y Documentación (UB 2007) - Bienio: 2002-2004. Tesis: 'Sistema matricial d'indicadors per a l'anàlisi estratègica de la informació a les organitzacions'. Licenciado en Documentación (UOC 2003). DEA 'Técnicas de análisis infométrica: de la inteligencia competitiva a la inteligencia estratégica' (UB 2004). Máster en Análisis económico aplicado (IDEC-UPF 2005). Ha sido profesor de Uso estratégico de la información, en la Universidad Jaume I de Castellón y de Gestión estratégica de la información en las organizaciones, en el Campus Universitario de la UPC. Actualmente es profesor propio de los estudios de Información y Comunicación de la UOC y profesor asociado de I+D y sistemas de información en el Departamento de Ciencias Económicas y Empresariales de la Universidad Internacional de Cataluña. También es licenciado en Filosofía y Ciencias de la Educación (UB 1992).

Lluís Anaya Torres

Jefe de Sistemas de Información de la Diputación de Tarragona. Profesor asociado del Departamento de Matemática e Informática de la URV y consultor de la UOC. Profesor del MBA de la URV.

Miquel Barceló García

Doctor en Informática, ingeniero aeronáutico y catedrático del Departamento de Lenguajes y Sistemas Informáticos de la Universidad Politécnica de Cataluña. Traductor y escritor, especializado en el género de ciencia ficción.

Ramon Costa i Pujol

Ingeniero en Informática y máster en Calificación pedagógica por la UPC y diplomado en Dirección General por EADA. Ha ocupado diferentes responsabilidades en l gerencia y dirección de proyectos en el ámbito de Sistemas de Información, Tecnología y Organización en diferentes empresas de consultoría e ingeniería del software, tanto locales como internacionales. Profesor asociado de Dirección de proyectos de la UPC y colaborador docente de las Escuelas Universitarias Gimbernat (adscritas a la UAB) y a la UOC en los Estudios de Ingeniería de Informática y Documentación. Vicedecano del Colegio Oficial de Ingeniería de Informática de Cataluña y colaborador de Radio Intereconomía.

Jaume Raventós Moret

Ingeniero químico por el Instituto Químico de Sarriá, ha desarrollado su carrera profesional como formador informático en empresas y centros de formación, y como consultor TIC diseñando y desarrollando aplicaciones de gestión y sistemas de bases de datos. Colabora con la UOC como consultor de la asignatura de Sistemas informáticos de los Estudios de Información y Documentación, desde el año 1999. Participa como profesor en varios programas formativos TIC en el Área de Promoción Económica del Ayuntamiento de l'Hospitalet de Llobregat.

Xavier Sánchez Porras

Licenciado en Filología Hispánica por la Universidad de Valencia. Especialista en Internet aplicado a la educación y a la empresa por la UNED. Consultor de la UOC desde el año 1999 en Multimedia y Comunicación, y Técnicas de Edición Electrónica. Autor de materiales didácticos por la UOC, ha realizado el diseño e implementación de diferentes proyectos nacionales e internacionales relacionados con los entornos virtuales, *e-learning* y Sistemas de Mantenimiento de Contenidos. Ponente en los cursos de formación del profesorado en TIC de los Centros de Profesores de Játiva y Gandia. Profesor de secundaria en el IES Veles e Vents del Grau de Gandia.

Ignasi Sebastià Oriol

Licenciado en Informática por la Universidad Politécnica de Cataluña. Principalmente dedicado a la docencia (universidades, empresas y escuelas) en proyectos relacionados con las TIC. Consultor de la asignatura de Sistemas informáticos de los Estudios de Ciencias de la Información de la UOC, a los que está vinculado desde el año 1997. Ha realizado diferentes proyectos nacionales e internacionales relacionados con los entornos virtuales, *e-learning* y *Content Management* System. Socio fundador de Marseb Consulting.

Jeroni Vivó i Plans

Licenciado en Ingeniería Técnica Superior de Telecomunicación por la Universidad Politécnica de Cataluña (UPC). Máster en Tecnologías de la información, posgraduado en Telemática, en Dirección de las telecomunicaciones y en Desarrollo de aplicaciones web en tecnología. NET. Ha desarrollado su carrera profesional como consultor de sistemas de información, desarrollando, diseñando e implementando proyectos para empresas del sector bancario, de las telecomunicaciones, farmacéutico y de la Administración pública. Colabora como consultor en la UOC desde 1998, es coautor del material docente de *Introducción a las tecnologías de la información* de los Estudios de Documentación.

Primera edición: febrero 2010

Av. Tibidabo, 39-43, 08035 Barcelona
Diseño: Manel Andreu
Realización editorial: Eureca Media, SL
ISBN: 978-84-692-8571-8
Depósito legal: B-2.418-2010

Introducción

La asignatura de *Tecnologías de la información* está dirigida a los estudiantes de Documentación, en su potencial papel de gestores de bibliotecas, centros de documentación y servicios de información diversos en todo tipo de organizaciones.

Se trata, como podéis ver si consultáis los objetivos generales de la asignatura, de adquirir fundamentos sólidos sobre las tecnologías de la información y de la comunicación. No se pretende que el estudiante se convierta en un especialista tecnológico en estos temas, pero sí que pueda tener una visión de la posible aplicación práctica de estas tecnologías con respecto a un servicio de información. Y, sobre todo, se desea que se pueda convertir en un interlocutor válido con los especialistas tecnológicos en situaciones en las cuales se puede encontrar, o quizás se ha encontrado ya, en su ejercicio profesional, como por ejemplo: evaluación de software para su posible adquisición, contratación de comunicaciones, diseño e implementación de bases de datos, etc.

El material didáctico empieza con un módulo dedicado al ordenador, a su arquitectura y sistemas operativos. El curso se inicia así, con el ordenador como elemento tecnológico más omnipresente y tangible. Y a continuación, se entra en un módulo dedicado al almacenaje de información y en particular en las bases de datos, unos aspectos tecnológicos que pueden resultar especialmente próximos e interesantes a los profesionales de la documentación.

En una segunda fase del curso se entra en dos temas de cariz más general, pero absolutamente imprescindibles para llenar de utilidad y contenidos nuestros ordenadores: por una parte, el software, sus características y aplicaciones. Por otra parte, las redes y comunicaciones como elemento tecnológico imprescindible, pero que demasiado a menudo pasa desapercibido.

Finalmente, se dedica un tema a los fundamentos de la inteligencia artificial, como elemento que puede ir más allá de las aplicaciones más clásicas de las tecnologías de la información, y suponer un salto cualitativo para potenciar las capacidades humanas individuales y colectivas.

El material dispone también de unos casos, situados en el módulo 6, que serán utilizados eventualmente en la acción docente y la evaluación continuada según las indicaciones de los consultores en cada semestre.

Éste, de forma muy resumida, es el recorrido de aprendizaje que estáis invitados a iniciar a continuación.

Objetivos

Con el uso de los materiales que conforman esta asignatura, conseguiréis los objetivos siguientes:

1. Adquirir unos fundamentos sólidos de tecnologías de la información y de la comunicación en los ámbitos de hardware, software y comunicaciones.

2. Conocer las posibilidades que brindan las tecnologías de la información y de la comunicación para su aplicación práctica en las organizaciones actuales.

3. Estar capacitado para ser un interlocutor válido para los especialistas en tecnologías de la información, en situaciones tales como: proyectos de gestión de información que implican la programación o instalación de software por parte de estos especialistas, necesidades que tenga un centro de documentación o un servicio de información de adquisición de software o de contratación de comunicaciones.

Contenidos

Módulo didáctico 6

Casos prácticos sistemas informáticos

Lluís Anaya Torres

1. Empresas de productos y empresas de servicios
2. Caso 1. Una administración: el Ayuntamiento de VilaUOC
3. Caso 2. Una empresa de fabricación de producto. MecanoUOC

Ordenadores y sistemas operativos

Xavier Sánchez Porras
Ignasi Sebastià Oriol

PID_00153116

UOC
Universitat Oberta de Catalunya
www.uoc.edu

Índice

Introducción

En este módulo didáctico podréis trabajar diferentes aspectos de los ordenadores y sus sistemas operativos. Se trata de distintas nociones, conceptos y cálculos que necesitaréis para poder afrontar la asignatura y comprender mejor el complejo mundo de los ordenadores, del software y del hardware.

En un primer momento, tendréis que reflexionar sobre la omnipresencia de los ordenadores en nuestras vidas y su necesidad en una sociedad moderna en todo el mundo donde los hay y donde tendrán que llegar.

Después, estudiaremos la importancia de la computación en la estructura industrial científica y empresarial de la sociedad. Veremos cuáles son los factores que han hecho posible y necesaria esta revolución de los ordenadores.

También repasaremos las aplicaciones más avanzadas de la computación hoy día, en el ámbito doméstico y el social. Y también la historia de los ordenadores desde los primitivos ábacos hasta los albores de la computación electrónica.

Ya más centrados en la lógica que en el hardware, veremos cómo la representación de las cantidades en sistema binario es una cuestión crucial para que se hayan desarrollado máquinas que calculan a una velocidad óptima. Esto lo haremos comparando con sistemas conocidos y comentando otros sistemas de numeración que son muy prácticos a la hora de aprender el funcionamiento y la programación del ordenador.

Provistos ya con los elementos necesarios entraremos en directo en un ordenador, de manera organizada; primero veremos cuáles son las partes que tiene sirviéndonos del modelo que concibió von Neuman y que ha servido de ejemplo hasta hoy.

Después veremos más detenidamente cada elemento en concreto: las memorias, su funcionamiento y tipo, los periféricos y su funcionalidad, las unidades de almacenamiento y sus diferentes tipos y ejemplos, además de su organización y funcionamiento. Para entender su intercomunicación, recurriremos a los estudios de Claude Shannon sobre la información y aprenderemos las confluencias entre la comunicación humana y la que realizan los ordenadores tanto internamente como externamente.

Con el concepto de información en la mano, nos adentraremos en su tratamiento y proceso, ejemplarizado en el Código Morse. Leeremos sobre **Boole** y su álgebra, que permitió concebir los mecanismos de funcionamiento de los

programas que corren en los ordenadores y cuya ejemplificación sirve también para saber un poco más sobre el misterioso "saber" de los ordenadores, relacionando la lógica de procesamiento con la lógica del pensamiento humano.

Ya en este punto, estamos preparados para conocer la parte más inteligente y amable de la máquina, el sistema operativo. Veremos cuáles son sus partes, sus funciones, sus distintas clasificaciones. Conoceremos la evolución de los sistemas operativos relacionada en primer lugar con las distintas generaciones de ordenadores, para pasar en último término a hacer un repaso por los principales sistemas operativos utilizados hoy día.

Dispondréis, finalmente, de dos apéndices que os permitirán lo siguiente: uno, saber cómo debéis pasar de la notación del sistema decimal a la notación del sistema binario; y el otro os enseñará la manera de hacer cálculos a partir de bits y bytes y os permitirá entender para siempre por qué vuestro módem ADSL de 256 kilobits funciona sólo a 32 kilobytes.

Objetivos

Los **objetivos generales** que el estudiante puede alcanzar son los siguientes:

1. Comprender el funcionamiento interno de un ordenador.

2. Introducir el álgebra de Boole y ver su relación con las puertas lógicas.

3. Familiarizarse con el análisis de pequeños circuitos lógicos.

4. Tener una visión de la función de os sistemas operativos.

5. Conocer diferentes maneras de clasificar los sistemas operativos.

6. Tener una visión general de la evolución de los sistemas operativos.

7. Conocer algunos de los sistemas operativos más significativos.

Estos objetivos generales se desglosan en los **objetivos específicos** siguientes:

1. Conocer que los ordenadores están por todas partes, nos acompañan y nos ayudan en nuestra vida diaria y en el trabajo.

2. Conocer y aplicar a los estudios documentales los conceptos de informática, ordenadores, software y hardware.

3. Identificar el impacto de los ordenadores en la sociedad.

4. Conocer aplicaciones de la informática

5. Conocer los orígenes de la computación y reflexionar sobre ellos.

6. Conocer los diferentes sistemas de representación y su relación con la computación.

7. Conocer la estructura de un ordenador y saber identificar cada una de sus partes.

8. Conocer la teoría de Shanon y Weaver.

9. Conocer la importancia que representa pasar de datos a información.

10. Conocer el proceso de comunicación con el ordenador.

11. Identificar diferentes soportes físicos que almacenen información.

12. Conocer un sistema matemático idóneo para el trabajo lógico con circuito electrónico.

13. Saber por qué un ordenador entiende un lenguaje escrito.

14. Conocer el concepto de sistema operativo.

15. Identificar las diferentes partes de un sistema operativo.

16. Conocer las clasificaciones de los sistemas operativos según diferentes criterios.

17. Conocer la evolución de los sistemas operativos.

18. Conocer los principales sistemas operativos de los últimos años.

1. Los ordenadores

1.1. Introducción: están por todas partes

En este apartado descubrimos que los ordenadores están por todas partes, nos acompañan y nos ayudan en nuestra vida diaria y en el trabajo. La película *Terminator* imagina un futuro en el que los ordenadores han llegado a sustituir al hombre e, incluso, intentan liquidar a la especie humana. Actualmente, estamos muy lejos de la ciencia ficción y, sin embargo, ¡estamos rodeados de ordenadores!

No hay ningún motivo de alarma, todos los ordenadores que nos rodean están a nuestro servicio, disfrutamos de los mismos y ganamos calidad de vida.

Por la mañana, nuestro despertador seguramente ya tiene algún microordenador incorporado que nos permite marcarle una hora para que nos despierte.

La ducha y el agua caliente de primera hora de la mañana nos las puede proporcionar una caldera que también regula la temperatura de la casa. En la cocina estamos rodeados de ordenadores: el ordenador que regula el frigorífico, el microondas, la vitrocerámica, la lavadora y el lavaplatos. Y además, dentro de poco, una disciplina emergente como la domótica hará que todos estos aparatos se puedan comunicar entre sí y con nosotros.

La domótica

Con la domótica, se consigue que las viviendas sean más confortables y seguras. Por ejemplo, podemos poner en marcha la calefacción sólo con enviar un mensaje SMS desde nuestro móvil.

Nos comunicaremos con las máquinas, desde lejos, por medio del móvil. El móvil también incorpora un microprocesador, y estos microprocesadores son cada día mejores y más complejos. El móvil, aparte de conectarse a Internet mediante WAP[1], GPRS[2] o UMTS[3], nos permitirá comunicarnos con las máquinas de bebidas, helados, etc.

(1)WAP (*wireless application protocol*, 'protocolo de aplicaciones sin hilos')

(2)GPRS (*general packet radio service*, 'radioservicio general de paquetes')

(3)UMTS (*universal mobile telecommunications system*, 'sistema universal de telecomunicaciones de móviles')

Sin embargo, el móvil lo tendremos que guardar cuando entramos en el automóvil para ir al trabajo, el coche nos indicará (y no hay que hablar del futuro, ya lo hacen determinados modelos actuales) si tenemos las puertas abiertas y los cinturones puestos o no, si hace frío o calor, si queremos que haga más frío o más calor, la gasolina que nos queda, la temperatura exterior, los niveles de líquidos, las medias de velocidad y de gasto, etc. A veces nos podemos sentir un poco como el pobre ciborg de la película.

Cuando llegamos al trabajo, a muchos nos espera lo que entendemos como un ordenador de mesa, con la pantalla expectante y el teclado preparado para que lo utilicemos.

Aunque, como hemos señalado, aquello que nosotros reconocemos como ordenador es el PC de nuestra mesa de trabajo, en realidad estamos rodeados: cada uno de los elementos que hemos enumerado en los párrafos anteriores lleva un chip en su interior que le permite hacer tareas repetitivas y ahorrarnos trabajo, pensar por nosotros las cuestiones que, aunque son automáticas y no requieren mucha inteligencia dedicada, sí que tendrían una dedicación si no tuviéramos un pequeño chip que hiciera estos procesos en nuestro lugar.

Estos chips de los electrodomésticos tienen una arquitectura cerrada, realizan siempre la misma tarea, que tiene unos caminos cerrados y finitos y se puede programar con facilidad.

Nuestro PC de mesa tiene una arquitectura abierta, puede realizar múltiples tareas y distintas funciones si tiene el software adecuado que hace funcionar la máquina en la dirección y con las operaciones requeridas.

Con estos dos modos (arquitectura abierta y arquitectura cerrada) no hay que ir muy lejos para encontrar estos aparatos por todo el mundo. Utilizamos el PC para conectarnos a Internet, generar, organizar y mantener todo tipo de datos y documentación, llevar las cuentas, leer el diario, jugar a videojuegos, y quizá menos para leer libros, dibujar, ver películas en DIVX[4] y montar el último vídeo del hijo o de la acampada en los Pirineos.

[(4)]DIVX es un formato de codificación de vídeo.

En el trabajo, la empresa mantiene los datos y la organización en una red de ordenadores PC y algún otro más grande que se interrelacionan, se comunican y trasvasan datos, crean facturas y procesan facturas, informes y memorandos que van a parar a bases de datos para que se consulten y vuelva a empezar el ciclo.

En la calle, centralitas inteligentes ordenan el tráfico, el alumbrado, el alcantarillado, la telefonía.

¿Cuánto tiempo es necesario para que nuestro coche, de acuerdo con el semáforo y la nevera y el horno de la cocina, sea capaz de calcular cuánto tardamos en llegar a casa y nos pueda tener el arroz en el horno preparado y humeante?

1.2. La revolución de los ordenadores en la sociedad

Aquí podéis ver cómo en los últimos siglos la tecnología ha hecho posibles los ordenadores, que hace tiempo que se concibieron y que son, al mismo tiempo, la base y el rasgo fundamental de la civilización actual.

La sociedad del siglo XXI ya es diferente. Aunque no hay que olvidar que esta sociedad de la que hablamos se reduce a determinados núcleos y, en conjunto, a un porcentaje minoritario de la humanidad, es un hecho que el modelo que tenemos del hombre del siglo XXI es el que hemos descrito en los párrafos anteriores.

El proceso ha sido fulminante. A finales del siglo XIX, sólo algunas casas tenían luz y, después de cien años, estamos rodeados de circuitos electrónicos que realizan gran parte de las tareas que nos resultan pesadas.

La primera revolución industrial se produjo en el último tercio del siglo XVIII, y se basó en la máquina de vapor, la hiladora, el proceso Cort en metalurgia y, en general, en la sustitución de las herramientas por las máquinas.

La segunda revolución se produjo a mediados de siglo XIX y se centró en la electricidad, el motor de combustión interna, la química basada en la ciencia, la fundidora de acero eficiente y la difusión del telégrafo y los inicios del teléfono.

El siglo XX, con los elementos de estas revoluciones anteriores ya incorporados en el funcionamiento y desarrollo de aspectos de la sociedad, ha disparado el proceso y lo ha acelerado. Hay que decir que las condiciones para la realización de una revolución suelen estar presentes en la sociedad mucho tiempo antes, hay un periodo entre la creación de las condiciones intelectuales y materiales de una tecnología y su puesta en marcha e incorporación a los usos sociales.

En el pasado siglo XX esta distancia entre las condiciones y la realización ha conseguido periodos cada vez más reducidos. Sólo tenemos que ver cómo cada seis meses, hoy día, podríamos cambiar nuestro PC de mesa por uno con el doble de potencia.

Por una parte, la electricidad -que ha significado la posibilidad de llevar la energía al lugar y en la cantidad que se necesita- ha sido un motor muy importante de este proceso. Por otra, el perfeccionamiento acelerado que han sufrido las diferentes tecnologías de comunicación, particularmente el **telégrafo**, el **teléfono** y medios de masas como la **televisión**, la **radio** e **Internet** han hecho que el conocimiento necesario para mejorar todo este proceso circule a gran velocidad y permita aumentar de manera exponencial las mentes y/o las máquinas que trabajan y/o colaboran en un mismo asunto.

Los primeros ordenadores estaban al servicio de los estados y sobre todo, por desgracia, de la guerra.

La tecnología y la guerra

Los presupuestos de defensa han colaborado extraordinariamente en la evolución de los ordenadores y por este motivo tenemos muchas de las tecnologías actuales. Sin embargo, sin tardar demasiado, todo aquello que estuvo en un principio vinculado a la ciencia y a la defensa se ha ido introduciendo en el sistema productivo y en la sociedad.

Los estados, ya lo hemos comentado, fueron los promotores de esta revolución de la computación. Después, la empresa y la industria empezaron a utilizarla para su organización y desarrollo en un primer momento y por último como producto final de su actividad.

Las empresas no sólo utilizaban las tecnologías para su funcionamiento interno, sino que poco a poco fueron creando productos tecnológicos que iban llegando a los usuarios finales, es decir, nosotros, que los consumimos. Los individuos nos empezamos a beneficiar de las comodidades que ofrecían estas tecnologías en último lugar.

La radio, el teléfono y el telégrafo mejoraron nuestras comunicaciones. La radio ya fue un medio de información importante durante el primer tercio del siglo XX, mientras que el cine y la televisión luchaban con la posibilidad de transmitir la voz junto con la imagen. En el momento en el que se pudo añadir voz a la imagen y se pudieron hacer accesibles los receptores al público en general, la televisión inició su camino. Detrás de la televisión vino Internet, que ya representa el instrumento más importante para el mundo de la comunicación.

El hecho de que se aplicara la misma tecnología a la comunicación hizo posible que el sistema se realimentara, cada vez había más invenciones y, al mismo tiempo, la relación entre los diferentes proyectos e invenciones que permitían los cada vez más ágiles sistemas de comunicación hacía que aparecieran nuevas posibilidades, de las cuales no se habrían dado cuenta si no hubiera existido la facilidad de comunicación comentada.

A partir del nacimiento de Internet, esto se ha convertido en exponencial, ha disparado la velocidad con la que la ciencia crea nuevos objetos y comunica sus descubrimientos a los otros científicos para que también puedan concebir nuevas creaciones, implementaciones o utilidades de las antiguas.

Los circuitos integrados y los transistores habían sustituido las viejas y aparatosas lámparas que hacían funcionar los ordenadores. Todos estos inventos que hemos enumerado más arriba ya funcionaron a partir de circuitos y transistores, lo cual ha permitido ir haciéndolos más adaptables y utilizables en diferentes situaciones y ha reducido su tamaño y adaptado su forma según las circunstancias y/o usos.

Las tecnologías inteligentes que nacieron dentro de los ordenadores se fueron exportando al entorno más próximo de los ciudadanos, incorporando chips inteligentes a las herramientas de uso cotidiano: electrodomésticos, vehículos, puestos de trabajo, reproductores musicales, máquinas de fotografiar, etc.

La industria y la empresa, grande y pequeña, han reestructurado todo su funcionamiento al utilizar los ordenadores, tanto en las oficinas como en las cadenas de producción o en las comunicaciones. De hecho, la misma producción de tecnología y la producción de contenidos para las tecnologías de comunicación se han convertido en sectores con un volumen muy importante de negocio.

Cotización en Bolsa

A pesar del hundimiento de la Bolsa que produjeron algunas empresas infladas, entre las que se cotizan mejor hoy día están CISCO o Sun, empresas centrales en el complicado mundo de las tecnologías de la comunicación; una crea el hardware que soporta el sistema (redes principales de Internet) y la otra, el software (Java).

En definitiva, hemos vivido una revolución que ha cambiado totalmente nuestro paisaje. Es muy probable que, si levantáis la vista de estos documentos, podáis contar al menos cinco o seis ordenadores que trabajan para vosotros.

¿Podrían tener razón los guionistas de *Terminator*?

1.2.1. Aplicaciones: los ordenadores omnipresentes

Las aplicaciones de los ordenadores y/o las tecnologías de la comunicación han tenido un gran impacto dentro del mundo industrial, empresarial, social, sanitario, cultural y, en general, sobre la calidad de vida. Los guionistas de *Terminator* no tenían razón, los ordenadores no tienen la capacidad de ser buenos ni malos, es el hombre quien los ha puesto a su servicio y, más allá de la ciencia ficción y de la tecnofobia, tenemos que pensar que el peligro no reside en las máquinas, sino en nosotros.

Hasta ahora estamos en paces. Si, por una parte, disponemos de una guerra *inteligente* y televisada, por la otra tenemos que ver que la ciencia se ha beneficiado de una manera extraordinaria de la potencia de cálculo de los ordenadores. Hoy día disponemos de programas distribuidos.

El proyecto SETI

En el proyecto SETI se permite utilizar los tiempos muertos de todos los ordenadores adscritos al proyecto -cualquiera se puede adscribir teniendo un PC y una conexión a Internet- para realizar cálculos que de otra manera serian imposibles (SETI: *search for extra-terrestrial intelligence*).

En medicina, no sólo se puede observar a los pacientes por dentro, lámina a lámina, de diferentes maneras, sino que, además, se puede realizar una operación a distancia con todas las ventajas que se pueden derivar de esto.

La enseñanza, cada día, incorpora poco a poco las nuevas tecnologías en las aulas con los ordenadores personales, la implementación de telecomunicaciones y los sistemas de proyección para las aulas. Hay unas pizarras sobre las cuales no sólo se puede proyectar el escritorio del sistema operativo Windows con todas sus posibilidades, sino que, además, el estudiante y el profesor pueden actuar sobre los iconos y las carpetas con el dedo e, incluso, dibujar como en las pizarras clásicas.

En el mundo empresarial, las telecomunicaciones y los ordenadores permiten mantener la información de todos los procesos actualizada al instante, la localización y la comunicación con las flotas de vehículos, comerciales y tiendas. Los empleados disponen de oficinas móviles basadas en ordenadores portátiles, *palms*, *pockets* o teléfonos móviles. Gracias a las tecnologías soportadas

Tercera edad y tecnología

Se piensa en la posibilidad de atender a personas de la tercera edad por medio de robots manejados desde otro lugar, incluso desde otros países.

por ordenadores grandes y pequeños la información fluye por dentro de la empresa y permite tomar decisiones cada vez más rápidamente y atacar los problemas en cuanto se presenten.

En domótica, se tienen aspiradores inteligentes capaces de recorrer la casa aspirando todos los rincones y, en su momento, volver para cargar energía de la red y después continuar sus tareas.

En el cine, la tecnología casi ha llegado a sustituir a los actores en las películas de Pixar o la misma *Final Fantasy*. Para el dibujo animado, para la creación de mundos en 3D[5], para los efectos especiales de las películas espectaculares e, incluso, de las menos espectaculares para mezclar unas y otras técnicas, los ordenadores son, han sido y serán fundamentales y centrales en el proceso. La televisión sigue los mismos pasos, con el añadido de que muchas veces las composiciones se hacen en directo sin posibilidad de rectificar, una tarea propia de ordenadores.

(5)3D es el sistema de representación visual que crea la sensación de tres dimensiones a partir de las dos de la pantalla.

Internet está tan presente que resulta inútil intentar dar una visión. Internet es, al mismo tiempo, el resultado de las tecnologías y el fomento de la comunidad que elabora estas tecnologías. No se puede pedir nada más.

Los viajes a la Luna y a Marte han sido posibles gracias a un conjunto innumerable de tecnologías.

Imágenes de Marte

Las mismas fotos que recibimos del planeta son una sofisticada tecnología que permite convertir un muestreo de las condiciones marcianas en una foto comprensible para el ojo humano, todo un éxito.

1.3. Computación, un problema antiguo

1.3.1. Los antiguos

El hecho de la computación ha sido un problema antiguo. En la antigüedad, el hombre le dio distintas soluciones, con tecnologías más rudimentarias que las de hoy. Desde siempre, cuantificar las cosas ha sido una necesidad del ser humano (cosechas, terreno, hitos, riquezas). Cuantificar representa un proceso metonímico por el cual abstraemos la cantidad de un objeto prescindiendo de los matices y detalles.

Una naranja y otra naranja nos dan matemáticamente dos naranjas, pero no tenemos que olvidar que si pesamos cada una individualmente u observamos su color y su forma no se trata de dos objetos iguales. Con el fin de igualarlos y, en consecuencia, cuantificarlos o computarlos, ha sido necesario abstraer uno de sus rasgos, la unidad. Tomamos una parte (la unidad) por el todo (la naranja), y por este motivo hablamos de *proceso sinecdótico* o *metonímico*.

Con los comentarios del profesor **Carl B. Boyer** en su *Historia de la matemática*, podemos ilustrar la cuestión:

"Este reconocimiento de la propiedad abstracta que tienen en común determinados grupos, la cual nosotros denominamos *número*, representa una etapa importante en el camino hacia la matemática moderna. Es completamente improbable que un descubrimiento como éste haya sido obra de un hombre individual ni de una única tribu: seguramente, debió de ser una especie de conciencia gradual que se podía haber producido dentro del desarrollo cultural humano tan pronto al menos como el uso del fuego, hace unos 400.000 años. Aquello que sugiere que el desarrollo del concepto de número fue efectivamente un proceso largo y lento es el hecho de que algunas lenguas, incluso el griego, han conservado en su gramática una distinción tripartita entre uno, dos y más de dos, mientras que la mayor parte de las lenguas actuales sólo hacen una distinción dual en el número gramatical entre singular y plural."

Así pues, a los antiguos no les fue difícil empezar a hacer una raya en el suelo o sencillamente con dedos de la mano para cuantificar sacos, cacharros o animales. Después, la computación resultaba muy sencilla, tantas rayas, tantos dedos, tantos elementos.

"La conciencia de número fue bastante extendida y clara para que se llegara a sentir la necesidad de expresar esta propiedad de alguna manera, al principio presumiblemente sólo en un lenguaje simbólico. Los dedos de la mano se pueden utilizar fácilmente para representar un conjunto de dos, tres, cuatro o cinco elementos, y no de uno porque el número uno, al principio, normalmente no se reconocía como un auténtico número. Por medio de dedos de las dos manos se podían representar colecciones de hasta diez elementos, y utilizando dedos de manos y pies se podía llegar hasta veinte."

"Cuando el uso de los dedos ya era inadecuado, se podían utilizar pequeños montones de piedras para representar una correspondencia biunívoca con los elementos de otro conjunto, y cuando el hombre primitivo utilizaba este sistema de presentación. A menudo amontonaba las piedras en grupos de cinco, por el hecho de que antes se había familiarizado con los quíntuples de objetos por observación de su propia mano o pie. Como hizo observar Aristóteles hace ya mucho tiempo, la extensión, hoy día, del uso del sistema decimal no es sino la consecuencia del accidente anatómico de que la mayor parte de nosotros nacemos con diez dedos en las manos y diez más en los pies."

De esta manera, y siguiendo estos estudios, vemos que el primer ordenador del hombre fueron sus propias manos; es evidente que no se trata de una máquina artificial, pero si se mira bien tiene determinadas ventajas.

También podemos observar aquí otro factor importante de esta cuestión: el hecho de que el sistema decimal se eligiera a raíz de un accidente natural, el número de nuestros dedos. Más adelante, trataremos el asunto de los sistemas de numeración en diferentes bases. Con el tiempo y la reflexión, el ser humano se fue separando de este sistema decimal a la hora de crear máquinas computadoras en favor de otros que tenían mejores prestaciones y adecuaciones al mecanismo, como el decimal.

Los egipcios conocían el uso del papiro, que les permitía hacer anotaciones de manera más fácil y flexible; cuando las anotaciones necesitaban ser permanentes, escogían la piedra que, claro está, se ha manifestado como un soporte realmente duradero, sobre todo si pensamos que la información archivada en nuestros CD y cintas VHS[6] morirá probablemente antes que nosotros mismos.

[6]VHS quiere decir *video home system*.

Puede ser difícil ver la semejanza entre el papel y un disco duro, CD o disquete, pero al fin y al cabo no son nada más que dos soportes que contienen información, codificada de alguna manera. Están hechos para ser leídos y escritos (ya sea por un hombre o por otro componente del ordenador -micro- que de alguna manera acabará relacionándose con un ser humano: teclado y pantalla).

En Mesopotamia escribían sobre tablillas de arcilla blanda a la hora de computar. Para hacer las muescas, utilizaban una varita en forma de cuña que dio origen al nombre de esta escritura *cuneiforme* (del *cunnus* latino).

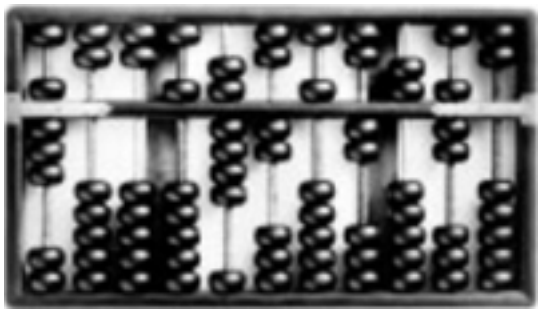
Las tablillas

La tablilla se cocía en un horno o se secaba al sol para que se fijaran las anotaciones; para decirlo en lenguaje moderno, el listón blando era RAM (*random access memory*) y, al cocerlo, lo convertían en ROM (*read only memory*).

Sin embargo, estos sistemas sólo tenían de computador el soporte, los cómputos todavía los tenía que hacer el ser humano, tanto escribirlos como realizarlos: la primera máquina de computar que ofrecía el resultado después de actuar más o menos mecánicamente fue el ábaco, inventado en torno al 3000 a. C. en Babilonia y mejorado más adelante hacia el 1300 a. C.

Tipos de ábacos

Hay muchos tipos diferentes de ábacos. Si por curiosidad o afición queréis ampliar el tema, podéis buscar imágenes en Google y consultar la Wikipedia.



Ábaco

1.3.2. La representación de las cantidades

El concepto de la representación de las cantidades es importante para comprender cómo puede "pensar" o "procesar" una máquina de computar. Todos hemos escrito números, hecho cuentas e, incluso, resuelto algún problema matemático alguna vez. Sin embargo, ¿nos hemos parado a pensar qué implica esto?

¿Por qué no escribimos sencillamente "uno más uno igual a dos" en lugar del habitual "1 + 1 = 2"? Por descontado, esta segunda manera es más corta, pero nadie sería capaz de leerla ni entenderla si no reconociera en la fórmula la oración "uno más uno son dos".

Se trata de dos representaciones diferentes de la misma cosa y, en consecuencia, equivalentes. No obstante, una es comprensible sin la otra, pero no al revés. Podríamos decir que "1" es una notación o dibujo breve que representa la escritura "uno"; la escritura "uno", por su parte, es un dibujo que representa el sonido de la lengua /uno/ que, a su vez, representa nuestra idea del uno o de la unidad.

Se trata de dejar claro que son cosas diferentes, la unidad que nosotros pensamos y sus representaciones, por medio de un sonido, un dibujo, escritura o notación; que la representación se puede hacer de múltiples maneras, pero ninguna altera la imagen mental de la que estamos hablando: cincuenta y seis se podrá escribir 56, LVI o cincuenta y seis, pero no dejará de ser la misma cantidad.

Las diferentes civilizaciones han elegido distintos sistemas de representación para escribir sus cantidades.

Los diferentes ábacos de los que hablábamos antes se ajustaban a las maneras de computar que tenía cada civilización que los utilizaba.

Observaremos nuestro sistema de representación, el que utilizamos habitualmente como punto de referencia, para después, en el apartado siguiente, describir otros que son muy utilizados en la informática.

Habitualmente, utilizamos el sistema decimal. Este sistema organiza los elementos primero en unidades "1", después éstas en decenas "10", las decenas en centenas "100", y así sucesivamente, "1.000", "10.000", "100.000". Ésta es la notación que se ha elegido porque es una metáfora visual de esta situación expuesta.

Sistema decimal

Colocamos las unidades en la parte derecha, 0001. Cuando las unidades llegan al nivel de agrupación siguiente, ponemos uno en las decenas 0010 y volvemos a contar las unidades 0011, 0012, 0013, etc.

Contamos de esta manera porque agrupamos los elementos de diez en diez. Cada diez del nivel anterior, añadimos 1 al nivel siguiente. Quizá por la antigua costumbre de contar con los dedos, agrupamos las cosas de diez en diez.

Por estas agrupaciones de diez en diez, denominamos a esta manera de contar *sistema decimal*. Sin embargo, sólo es una manera de representar la información. Se puede hacer de muchas otras maneras sin alterar, en ningún caso, las cantidades de referencia. Podríamos escoger cualquier otra manera de agrupar las cuentas y llegaríamos a la misma situación. Por ejemplo, podríamos reunirlas a conveniencia en grupos de dos, de ocho o de dieciséis si nos interesara por cualquier razón.

1.3.3. Representación binaria, octal y hexadecimal

En este apartado, veremos distintas aplicaciones de la representación de cantidades. Hemos escogido las representaciones más habituales en el mundo de la informática en comparación con el sistema decimal, que es el más extendido en la vida diaria.

La representación binaria

La menor agrupación que podemos hacer es un grupo de dos, ya que el sentido común nos dice que la unidad no es grupo, así que podríamos ir agrupando las cuentas en grupos de dos. Cuando anteriormente lo hacíamos en el sistema decimal, el procedimiento consistía en colocar una notación en la celda de la izquierda cada vez que una agrupación se cumplía, y al llegar a diez elementos en la unidad colocábamos "10".

Pues bien, con el objetivo de representar cantidades en numeración binaria, haremos exactamente lo mismo:

	binario	**decimal**
Imaginemos una matriz de cuatro cifras.	0000	0
Sumamos una unidad.	0001	1
Sumamos otra unidad, ahora ya tenemos dos y el grupo ya está completo.	0010	2
Sumamos otra unidad.	0011	3
Sumamos otra unidad, de nuevo el grupo ya está completo, pero al aumentar el de su izquierda también lo completa.	0100	4
Sumamos otra unidad.	0101	5

Y así en lo sucesivo.

A simple vista, parece un sistema complicado y poco útil; para quien ha trabajado toda la vida con el decimal resulta realmente extraño, ¿qué ventajas tiene? Porque, precisamente, su sencillez binaria es la que le aporta toda la fuerza y la conexión o paralelismo con el ordenador.

Nosotros, culturalmente, hemos heredado el sistema decimal y nos parece tan natural como nuestra lengua, nos resulta sencillo si lo hemos aprendido de jóvenes y practicado en la escuela o a los negocios. No obstante, como código no es tan sencillo como podría ser o como sería deseable.

Un código es una correspondencia biunívoca entre dos conjuntos, cada elemento del uno -cantidad o concepto- es representado por un elemento del otro -su notación.

En sistema decimal, necesitaremos diez signos diferentes para anotar las cantidades.

Uno	1		0001
Dos	2		0010
Tres	3		0011
Cuatro	4		0100
Cinco	5		0101
Seis	6		0110
Siete	7		0111
Ocho	8		1000
Nueve	9		1001
Diez	1+0		1010

Observad cómo con la notación en el sistema binario únicamente se han utilizado dos signos (1 y 0), mientras que en la notación numérica hemos necesitado diez y en la escritura habitual, gran parte del alfabeto.

En la tabla anterior se ve con claridad la ventaja (por economía) de la codificación binaria comparando la escritura alfabética, el sistema de representación decimal y el binario:

- Con el fin de denominar las cantidades con palabras necesitaremos gran parte del alfabeto (u, n, o, d, s, t, r, q, a, c, i, e, v), organizado en agrupaciones totalmente arbitrarias como son las del lenguaje natural.

- Para hacerlo, en el sistema decimal utilizamos hasta diez símbolos o dibujos (0, 1, 2, 3, 4, 5, 6, 7, 8, 9), organizados en el sistema posicional, que

tiene posiciones infinitas teóricamente, siempre que no estemos limitados por el soporte para las posiciones: el lugar donde se quieran hacer las anotaciones es hasta un punto determinante, ya que limitará la anchura de la palabra (cifra que hay que escribir) que contendrá la representación.

- Para el caso del sistema binario, de aquí su gran ventaja, sólo necesitaremos dos signos (1 y 0; + y -), también organizados en un sistema posicional, como el decimal. Aunque las reglas de manejo ofrecen la misma complejidad que el decimal, no obstante, sólo necesitaremos representar dos signos. Esto permitirá utilizar mecanismos de representación muy simples: abierto/cerrado, luz/oscuridad, electricidad/no electricidad, sonido/silencio. Estas representaciones son adecuadas para representar los estados de una máquina mecánica o electrónica.

Reanudemos ahora un concepto que acabamos de esbozar, el de "palabra", y también nosotros entendemos por lo común el término *palabra* como un conjunto de sonidos entre dos silencios o un conjunto de grafías entre dos espacios blancos. Ahora utilizaremos este término para designar la anchura de celdas o posiciones que sea capaz de representar el mecanismo del que estemos tratando.

Si hablamos de un papel, las notaciones no podrán sobrepasar la anchura del papel (aunque en teoría podríamos ir añadiendo papeles uno al lado del otro, resultaría, de hecho, un procedimiento muy poco práctico); si pensamos en un ábaco, las posiciones se representarían en cada fila (cuidado, no en cada piedra, las piedras de una misma fila serían las unidades de esta posición), de manera que el ábaco incorporaría tantas posiciones como filas de piedras tiene.

A este número de posiciones le denominaremos *anchura de palabra*. La anchura de palabra es importante porque determina el número máximo de signos que podremos generar.

Con una anchura de uno, podemos elaborar dos signos.	0,1	(2)
Con una anchura de dos, podemos elaborar cuatro signos.	00, 01, 10, 11	$(2)^2$=2*2
Con una anchura de tres, podemos elaborar ocho signos.	000, 001, 010, 011, 100, 101, 110, 111	$(2)^3$=2*2*2
De esta manera, no resulta difícil continuar: con cuatro cifras podemos obtener... (haced el ejercicio).	(Haced el ejercicio.)	$(2)^4$ = 2 * 2 * 2 * 2

El ábaco binario

De esta manera, un ábaco binario (ideal) se diferenciaría de un decimal (ideal) con las mismas posiciones por el número de piedras en cada fila:

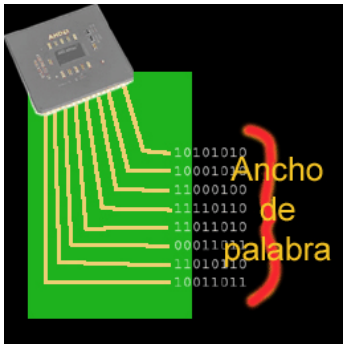
Como habréis observado, y siempre hablando en términos binarios, para obtener el número máximo de signos deberemos multiplicar las posibilidades de cada posición (2) por las posibilidades (2) del resto de las posiciones. En esto tampoco se diferencia del decimal, en el que tres posiciones permitirían 10 * 10 * 10 = 10^3 = 1.000 (000 - 999) signos o notaciones.

Más información

En **el apéndice I** encontraréis la manera de transcribir signos de notación binaria a decimal.

No podemos dejar este apartado sin que quede clara la relación profunda entre el ordenador y la notación y numeración binaria. Se trata de una caso de adecuación -aunque, evidentemente, no es casual sino fruto del estudio y la ingeniería- del código (elementos y reglas) al soporte que lo manejará.

Los ordenadores actuales, como podemos comprobar, funcionan a partir de transistores, y los transistores presentan dos estados, activado (1) o desactivado (0): un soporte perfecto para escribir y leer con, y tan sólo con, unos y ceros.



Transistores

También se debe tener en cuenta que quien **lee** y **escribe** propiamente dentro de un ordenador es el microprocesador, y este elemento es un aparato finito con una mirada limitada, no puede observar toda la información al mismo tiempo, lo tiene que hacer en la medida de sus ojos, unos pivotes de cobre que sólo distinguen si hay señal (1) o no la hay (0) y, a su vez, sólo son capaces de generar (1) o no (0) señal con cada pivote; ojos/boca (entrada/salida) pobres pero eficientes, que están limitados por su número, tantos pivotes, tantas lecturas a un tiempo, a una determinada anchura de palabra, una determinada limitación del número de unos y ceros que se tienen que recibir o crear.

Así pues, en un microprocesador con ocho entradas simultáneas se dice que tiene una palabra de 8 bits (cada bit es una unidad de entrada/salida que puede tener dos estados 0 y 1, y coincidiendo con el sistema de agrupación binario, hay un grupo de dos estados posibles).

Número de posiciones para generar números binarios

Un ordenador de ocho bits tiene ocho posiciones para generar números binarios.

Esta palabra marca el alcance de la mirada del microprocesador, determina todas las entradas/salidas que lee/escribe a un tiempo y, al mismo tiempo, el número máximo de signos o notaciones que se pueden utilizar como código de entendimiento (el microprocesador, como veremos más adelante, tiene que enfrentar todo tipo de entradas y salidas, se debe comunicar con una infinidad de periféricos). Si aplicamos el cálculo que hemos aprendido anteriormente, hay ocho posiciones capaces de contener dos signos:

Número de signos diferentes 2^8 = 2 * 2 * 2 * 2 * 2 * 2 * 2 * 2 = 256 signos diferentes que puede gestionar el ordenador en formato binario..

La representación octal

La representación octal sigue las mismas pautas que la que hemos visto hasta ahora. Sus signos serían 0, 1, 2, 3, 4, 5, 6, 7. La diferencia consistiría en el hecho de que después del 7 viene el 10. Si nos fijamos en las notaciones, son los mismos que los decimales pero el significado varía 10 en octal, y corresponde a la cantidad que nosotros entendemos como 8 en decimal.

Se reconocen las representaciones octales para llevar un cero a la cifra más a la izquierda 0456 frente al decimal 456.

La representación hexadecimal

Otra forma de representación numérica muy utilizada, junto con las que hemos visto antes, decimal, binaria y octal, es el hexadecimal. La hexadecimal concibe las cantidades organizadas en grupos de dieciséis, cosa que enseguida implica un problema si hemos comprendido lo anterior. Para los sistemas que hemos visto hasta ahora, siempre podíamos utilizar las notaciones 0-9 del sistema decimal, porque siempre eran códigos que tenían menos de diez elementos. Sin embargo, ¿qué pasa ahora que tenemos más de diez? ¿Cómo supliremos la carencia de notaciones entre 9 y 16?

Muy sencillo, introducimos letras -también notaciones- del alfabeto habitual para indicar los números que nos faltarán en las unidades de la manera siguiente:

0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
0	1	2	3	4	5	6	7	8	9	A	B	D	E	F	

Hay que indicar que igual que en el octal se reconocía la base para disponer de un cero en la parte izquierda, en el hexadecimal se ha elegido la representación 0x ante los dígitos para indicar la base hexadecimal; de este modo, 0xff significa, tal y como podemos comprobar en el apéndice I:

P1	**P0**		**Posiciones = 2(Px)**
16	1		**Base = 16**
16^1=16	16^0=1	Total	**Valor = 16^n**
f * 16	**f** * 1	= 255	**Notación = 0xff**
(15 * 16) +	(15 * 1) +	= 255	

Equivalencias

Veamos unos ejemplos de equivalencias entre los diferentes sistemas de numeración que hemos estudiado:

Decimal	Binario	Octal	Hexadecimal
15	1111	17	F
84	1010100	124	54
219	11011011	333	DB

1.3.4. Máquinas de computar hasta el ordenador electrónico

Las máquinas de computar y otras invenciones relacionadas que precedieron la aparición del primer ordenador electrónico representan un largo recorrido histórico. A causa de la extensión y el gran número de máquinas de computar que podemos exponer aquí, hemos preferido ofrecer un resumen ordenado por siglos y eligiendo sólo los fenómenos que afectan a los ordenadores. Hemos prescindido a propósito de la parcialidad que puede ofrecer esta visión, de aquello que, estrictamente, no tenga nada que ver con el ordenador físico. Más adelante, en la parte correspondiente a los sistemas operativos, hablaremos de lo que hace referencia al software y la gestión de la máquina:

Siglo XVII	
1617	**John Napier** (1550-1617) creó los rodillos de Napier, que multiplicaban basándose en sumas; probablemente, copió la idea de los árabes.
1621	**William Oughtred** (1574-1660) y **Edmund Gunter** (1581-1620) inventan la primera regla de cálculo.
1623/24	**Wilhelm Schickard** (1592-1635) inventó una calculadora que era una combinación de las máquinas de Napier y de Pascal, y que se denominó *reloj de calcular*.
1642	**Blas Pascal** (1623-1662) crea su *machina arithmetica*, *Pascalina*.
1671	El filósofo y matemático alemán **Gottfried Wilhelm Von Leibniz** (1646-1716) fue quien construyó la **primera calculadora**, con un tambor cilíndrico denominado *rueda escalonada* que no desaparecería de las calculadoras que lo sucedieron.
Siglo XVIII	
1793	**Chappe** (1763-1805) crea en Francia un telégrafo óptico.
1800	**Alejandro Volta** inventa la pila eléctrica.
Siglo XIX	
1812/22	**Carlos Babbage** (1792-1871) concibió la máquina diferencial, para la Royal Astronomical Society, **instrumento mecánico para calcular e imprimir tablas de funciones**.
1820	**Charles Xavier Thomas Colmar** inventó el aritmómetro, que llegó a ser la primera calculadora producida en serie.

1832	**Charles Babbage** desarrolló la máquina analítica. Era un **ordenador mecánico de propósito general**, para cualquier tipo de cálculo, lo cual presuponía el uso de **un programa**. Memorizaba mil números de cincuenta cifras, disponía de una **unidad aritmética lógica para los cálculos**, una **unidad de control** para establecer la orden de las operaciones, una **lectora de fichas perforadas** como entrada y una "**impresora**" para la salida de los resultados. Funcionaba con vapor como fuente de energía. Sumaba en tres segundos y multiplicaba en cuatro. **Ya era realmente un ordenador en termas conceptuales e incorporaba los componentes básicos de la programación.**
1837	**William F.Cooke** y **Charles Wheatstone**, junto con Morse, patentan el telégrafo eléctrico, basándose en la sensibilidad del imán a la electricidad.
1843	**Pehr Georg Scheutz** (1785-1873) y **Eduard Scheutz** construyeron un prototipo funcional de la máquina de Babbage.
1870	**G. R. Carei**, juntando las propiedades del selenio y los procedimientos de las primeras fotografías, diseña un sistema televisor rudimentario.
1872	**Frank Baldwin** crea una máquina que fue la antecesora de la **calculadora** de mesa.
1876	**Alexander Graham Bell** (1847-1922) desarrolló un aparato que podía transmitir sonidos por medio de un cable. Las comunicaciones habían sufrido una revolución.
1878	**Willgodt Theophil Odhner** (1845-1905) patentó en Rusia la Brunsvinga, que se empezó a producir en serie y se comercializó por toda Europa, hasta 20.000 unidades.
1879	**James Ritty** inventa la popular **caja registradora**, derivada de la de Frank Baldwin.
1884	**Herman Hollerit** (1860-1929), que trabajaba para la oficina del censo de Estados Unidos, **diseñó una tarjeta para tomar los datos de los empadronados: se perforaba** y después, con unas agujas y un recipiente con mercurio, se conseguía que hicieran contacto o no según si estaban perforadas o no. **Una tabuladora registraba los datos a doscientas tarjetas por minuto**, y así consiguió multiplicar por cien la velocidad del proceso y obtener el censo mucho antes de lo que se había conseguido anteriormente (diez años). En 1896 **fundó Tabulating Machine Co.**, que se transformaría en **Compunting Tabulating Recording, predecesora de IBM**.
1884	**William Burroughts** crea una sumadora provista de impresora derivada de la de Frank Baldwin.
1885	**Dorr Eugene** inventa la **primera calculadora mecánica que incorpora teclado y consigue sumar, restar, multiplicar y dividir** a partir de las operaciones anteriores, el computómetro.
1888	**León Tire** crea la **primera calculadora con mecanismo multiplicativo** a partir de mecanizar el uso del ábaco.
1891	**Otto Schaffer** crea una máquina basada en el Hollerit para el censo del imperio austrohúngaro.
1893	**Otto Stieger** (1858-1923) desarrolla una calculadora mecánica que permite la multiplicación directa a partir de una manivela, se denomina la *Millonaria*.
1899	**William S. Burroughs** crea una calculadora que multiplica, con teclado y dispositivo de impresión.
	Siglo xx
1901	**Marconi** inventa la radio. La primera emisión tuvo lugar en 1906 en Estados Unidos.

1907	**Ludwig Spitz** presenta una máquina, *Time is Money*, que podía sumar, restar, multiplicar y dividir.
1920	**Leonardo Torres Quevedo** (1852-1936), ingeniero, creó un aritmómetro en el que **introdujo la aritmética de punto flotante, como la que se utiliza hoy día**.
1925	**John Baird** inventa, por fin, la televisión.
1930	En los albores de la electrónica, **Vannev Bush** construye la última **calculadora analógica mecánica**, capaz de resolver ecuaciones diferenciales.
1937	**George Robert Stibitz** desarrolla la **primera calculadora digital basada en relés**, *Modelo K*, aunque no era programable.

1.4. Los ordenadores: una solución moderna

1.4.1. Máquina de Von Neumann

En este apartado, veremos el primer diseño conceptual de la arquitectura del ordenador que, después, llevado a cabo, representó la revolución que vivimos hoy día. En el año 1946, el matemático J. Von Neumann, en colaboración con Arthur W. Burks y Herman H. Goldstine, escribió el artículo "Preliminary discussion of the logical design of an electronic computing instrument", en el que se estableció el diseño de la arquitectura de un ordenador. Este diseño ha perdurado en el tiempo y, de hecho, todas las arquitecturas de ordenadores aparecidas hasta hoy día parten de este diseño.

Preliminary discussion of the logical design of an electronic computing instrument

El artículo "Preliminary discussion of the logical design of an electronic computing instrument" se puede encontrar en línea y en **Arthur W. Burks; Herman H. Goldstine; John von neumann** (1946, 28 de junio). "Preliminary discussion of the logical design of an electronic computing instrument" (2.ª ed., 2 de septiembre de 1947, 42 págs.). Princeton, Nueva Jersey: Inst. for Advanced Study [reeditado en *Yon Neumann's Collected Works* (1963, vol. 5, A. H. Taub Ed., págs. 34-79). Londres: Pergamon].

Fundamentalmente, con esta máquina se puede leer o ejecutar un conjunto de órdenes de instrucciones máquina que se encuentran, de alguna manera, dentro de la memoria del ordenador.

Intentaremos explicar las diferentes partes del ordenador siguiendo un ejemplo de la vida real, pues a veces esto nos ayuda a entender mejor los conceptos o, al menos, a centrar el tema. A medida que comentamos algún concepto nuevo, haremos una analogía con respecto a lo que sería nuestro ejemplo.

El cocinero inexperto

Se trata de un cocinero inexperto que quiere hacer un pastel y, para conseguirlo, utiliza un libro en el que se le indican los ingredientes necesarios, los electrodomésticos que tiene que utilizar, los pasos que debe seguir, etc.

El diseño de este modelo está formado por tres partes.

1) Memoria principal: es un espacio donde se almacena temporalmente la información del ordenador, es decir, los datos y las instrucciones. Von Neumann ya estableció que habría, dentro de la memoria principal, dos partes diferenciadas: **la memoria permanente**, donde encontraríamos los datos o instrucciones que utilizamos con frecuencia, y **la memoria de trabajo**, donde encontraríamos los datos o instrucciones que se utilizan esporádicamente.

Ejemplo de contenidos en la memoria principal

Supongamos que estamos escribiendo un texto en el ordenador. Según el diseño de Von Neumann, tendríamos en la memoria permanente el sistema operativo (Windows, Mac OS, Linux, etc.) y en la memoria de trabajo, el programa procesador de texto y el texto.

El cocinero inexperto (continuación)

En el ejemplo de nuestro cocinero, la memoria principal sería la mesa donde se dejan los platos, unos de éstos con ingredientes necesarios para hacer el pastel, otros vacíos que se deben llenar después de hacer alguna manipulación de alimentos, un atril para apoyarse el libro donde se indican los pasos que hay que seguir, etc. Evidentemente, podemos decir que nuestro cocinero es una persona muy ordenada: cuando tiene que ir a buscar un ingrediente, sabe dónde está la mesa.

2) Unidad central de proceso (UCP) o *central processing unit* (CPU): esta parte se encarga de ejecutar o llevar a cabo las órdenes o instrucciones de los programas.

También se le denomina *procesador* o *microprocesador* (por su tamaño).

La UCP también se divide en la **unidad aritmética y lógica** (es la que realiza la ejecución de la instrucción) y la **unidad de control** (la que coordina cómo y cuándo se debe hacer esta ejecución).

Si tuviéramos que decidir cuál es la parte más importante del ordenador, diríamos que se trata de la unidad central de proceso, aunque, evidentemente, sin la memoria ni los periféricos no tendríamos ordenador.

El cocinero inexperto (continuación)

En este caso, la unidad aritmética y lógica es el libro del cocinero. Cada paso de la elaboración del pastel se explica con todo detenimiento y con orden. El cocinero sería la unidad de control; aunque es el libro el que indica los pasos que se tienen que seguir, el cocinero da las señales necesarias para que se realicen los pasos del libro.

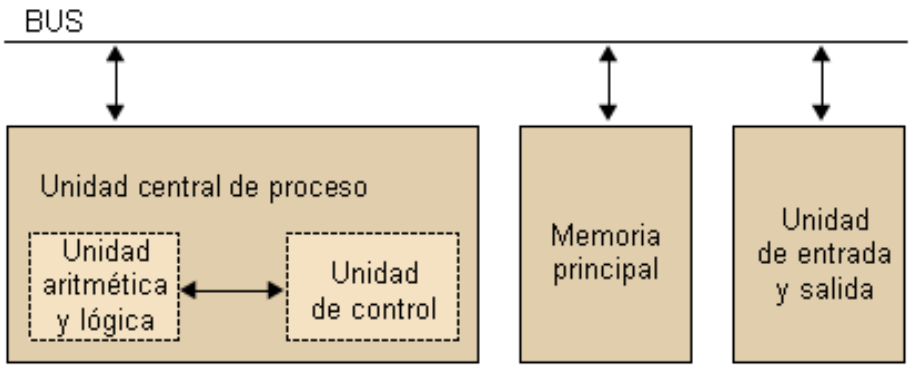
3) Unidad de entrada/salida (E/S): esta parte se encarga de comunicar los datos y las instrucciones de la UCP + memoria principal con el mundo exterior. Según la dirección en la que viajen los datos, hablaremos de periféricos de entrada (E), de salida (S) o de entrada y salida (E/S).

El cocinero inexperto (continuación)

Los diferentes robots de cocina o pequeños electrodomésticos serían los periféricos. Un microondas por sí solo sabe qué debe hacer. Sólo necesita que alguien le pulse el botón.

Cualquier dato u orden que tenga que viajar desde una parte hasta otra de este diseño de ordenador lo hará mediante unas carreteras denominadas **buses**.

En la figura siguiente, podemos observar la arquitectura del ordenador digital propuesta por Von Neumann:



- Los ordenadores actuales se basan en la arquitectura de von Neumann.
- Las partes principales de la arquitectura son unidad central de proceso, memoria principal y unidad de entrada/salida. Se interconectan mediante un bus.

1.4.2. Estructura básica de un ordenador

Pasemos ahora a ver no ya un modelo ideal de máquina ordenadora y comunicadora, sino los elementos que componen hoy día un hardware de ordenador. A pesar de que la arquitectura de los ordenadores actuales parte del diseño de von Neumann, estos ordenadores han sufrido algunas modificaciones con el fin de ganar prestaciones y adecuarse a los adelantos tecnológicos. Más adelante, hablaremos de algunos ejemplos de estas nuevas arquitecturas de ordenadores.

Para tener una visión más amplia del funcionamiento de un ordenador, profundizaremos en cada una de sus partes, y nos mantendremos fieles al diseño original:

1) La **memoria principal** (MP), donde se almacenan temporalmente datos e instrucciones, se divide en muchas celdas del mismo tamaño que se pueden identificar por una dirección única. Sería como hablar de un armario muy grande lleno de cajones distribuidos en filas y columnas. Todos los cajones tendrían el mismo tamaño y, con el fin de identificarlos, se debería indicar la

fila y la columna donde están. De esta manera, el cajón de la fila X, columna Y, sólo podría ser uno y se evitaría cualquier ambigüedad. Cuanto mayor sea el armario, mayor sería nuestra memoria.

La gran ventaja de la memoria principal es la velocidad y el gran inconveniente, si lo podemos decir así, sería la volatilidad, es decir, necesita suministro eléctrico para funcionar.

Pérdida de datos

Si estoy escribiendo un texto en el ordenador y se corta el suministro eléctrico, perderé todos los datos si previamente no he guardado el documento con la opción correspondiente de la aplicación.

El tiempo de acceso a la memoria se mide en nanosegundos (10^{-9} segundos).

¿Cuál es el funcionamiento interno de la memoria?

El funcionamiento de estas unidades de memoria es muy simple: la unidad central de proceso (UCP) le entrega o le pide un dato o instrucción que está en una dirección determinada de la memoria.

Por ejemplo, para leer el dato que hay en la dirección 8, se activará la señal de lectura, la dirección de memoria sería la 8 y el retorno sería el contenido de la celda 8.

La memoria principal

La memoria principal se confunde a menudo con el disco duro, aunque no tienen nada que ver. Hablamos de *memoria del disco duro* cuando tendríamos que decir *capacidad del disco duro*.

Hasta ahora hemos utilizado el término *memoria principal* de manera genérica, pero los ordenadores trabajan principalmente con dos tipos de memorias: la ROM[7] y la RAM[8]. La principal diferencia es que la ROM no permite escribir (al menos inicialmente).

[7]La ROM (*read only memory*) sólo es memoria de lectura.

[8]La RAM (*random access memory*) es memoria de acceso aleatorio.

La ROM se utiliza para almacenar información que no cambia con el tiempo (cosa que hace esporádicamente). La memoria principal utiliza la memoria RAM, aunque también podemos encontrar memoria RAM en otras partes del ordenador y con finalidades diferentes.

Hay otra memoria adicional y que complementa la memoria principal que se denomina *memoria caché*. Esta memoria es muy rápida, no tiene mucha capacidad pero ahorra determinados accesos a la memoria principal cuando buscamos informaciones repetidas en poco tiempo. En definitiva, se consigue aumentar la velocidad del ordenador.

¿Cuánta memoria principal puede tener un ordenador?

En principio, podríamos pensar que cuanta más memoria principal tengamos en nuestro ordenador mejor, pero en realidad esto no siempre es así. Hay una característica de la CPU que determina la cantidad máxima que se permite instalar. Mientras no se llegue a este límite, podremos ampliar.

2) La **unidad aritmética y lógica** (UAL) se encarga de realizar las operaciones aritméticas y lógicas. Está compuesta por un conjunto de circuitos digitales que permiten realizar un conjunto de operaciones determinadas, como sumar, restar, desplazar un bit a la izquierda, operaciones lógicas (AND, OR, NOT), etc.

Normalmente, se reciben los datos de la memoria, se realiza la operación y el resultado se vuelve a depositar en la memoria principal. Para evitar estas idas y venidas, y para ganar velocidad en el proceso de cálculo, la UAL tiene un conjunto de **registros** que le permiten almacenar temporalmente estos resultados. Normalmente, un ordenador actual no tiene más de 128 registros.

3) La **unidad de control** (UC) se encarga de coordinar que las ejecuciones de las instrucciones en la unidad aritmética y lógicos se realicen convenientemente. Para hacerlo, tiene un conjunto de señales de control.

Podríamos decir que la UAL realiza las ejecuciones de órdenes en el momento en el que la unidad de control le indique, además, que se encargue de coordinar, mediante señales, que cuando se ha ejecutado una instrucción la UAL esté preparada para la próxima ejecución.

Por ejemplo, para hacer una suma de dos valores que hay en la memoria principal, primero tendría que activar la señal de lectura de las dos posiciones de memoria, dejar pasar estos valores hacia la UAL y activar la señal para realizar la operación suma. Como os podéis imaginar, no podríamos iniciar la operación de la suma sin haber recibido los operandos diferentes.

La unidad de control

La unidad de control lleva internamente una especie de reloj que le va indicando el ritmo que tiene que llevar en estas tareas de coordinación. Pongamos un ejemplo para intentar ayudar a entender este concepto: sería muy parecido a las personas encargadas de tocar el tambor para marcar el ritmo dentro de las galeras de los barcos romanos. A cada golpe de tambor, los "esclavos" tenían que hacer un estirón de brazos con los remos.

Dentro de esta parte hay un componente, el **contador de programas** (*program counter*), que se encarga de indicar cuál es la instrucción siguiente que se tendrá que ejecutar en cada momento. Podríamos pensar que cuando se ejecuta un programa en un ordenador, las instrucciones se han ejecutado secuencialmente, una detrás de la otra, pero esto no siempre es así y, con mucha frecuencia, se producen saltos. Esto hace que este registro sea muy relevante en todo el proceso de ejecución de programas que, en definitiva, es la única cosa que sabe hacer un ordenador.

4) La **unidad de entrada y salida** (UE/S) se encarga de coordinar la comunicación de la unidad central de proceso con lo que sería el mundo exterior. Estas unidades exteriores reciben el nombre de **periféricos** y, según el sentido en el que viaje la información, hablaremos de **periféricos de entrada** (E), **de salida** (S) y **de entrada y salida** (E/S).

¿Cómo podemos saber de qué tipo es cada periférico?

Nos tenemos que imaginar qué relación hay entre la CPU y el periférico en cuestión; hacia dónde viaja la información útil que los relaciona.

Por ejemplo, para saber qué tipo de periférico es la impresora, podemos seguir la figura siguiente:

Si la CPU *da* información útil a la impresora, diremos que se trata de un *periférico de salida*, mientras que si la CPU *recibe* información útil de la impresora, diremos que se trata de un *periférico de entrada*. Si se da el caso de que la CPU *recibe y da* información de la impresora, diremos que se trata de un *periférico de entrada y salida*.

Periféricos

Podéis observar la importancia del verbo para saber de qué tipo de periférico tratamos: **recibir** o **dar**.

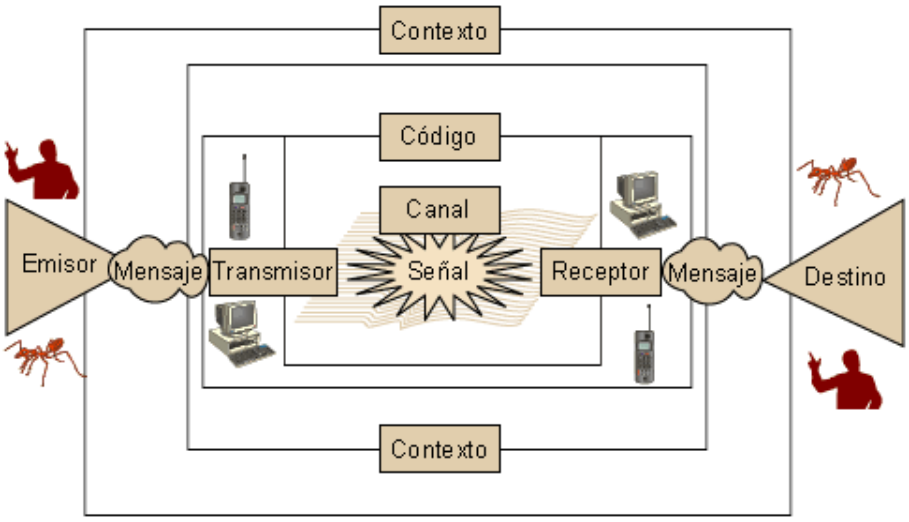
En nuestro ejemplo, la CPU sólo le da información útil a la impresora. Por lo tanto, diremos que la impresora es un periférico de salida.

- La unidad de memoria principal (UMP) guarda temporalmente datos e instrucciones.
- La unidad aritmeticológica (UAL) realiza las operaciones aritméticas y lógicas.
- La unidad de control (UC) coordina la ejecución de las instrucciones en las unidades aritméticas y lógicas.
- La unidad de entrada/salida (UE/S) se encarga de comunicar la unidad central de proceso con elementos exteriores (los denominados *periféricos*).

1.4.3. La información en la teoría de la comunicación de Claude Shannon

Se habla de la teoría de la comunicación establecida por Shannon con la finalidad de determinar cómo se tienen que comunicar tanto los elementos de un ordenador como los ordenadores o hardware entre sí o con los seres humanos. Por este motivo, se presenta a continuación un esquema general de sistema de comunicación. Trataremos más ampliamente estos conceptos en el módulo Redes y comunicaciones.

La información se transporta siempre en un proceso de comunicación. Un **emisor** comunica un mensaje (información) a un **destinatario**, por medio de un **transmisor** que está unido a un **receptor** por un **canal** que sufre modificaciones (**señales**) al trasladar el **mensaje** (la información).



Interferencias

Existe la posibilidad de que en el proceso de comunicación se hayan producido interferencias (ruido en sentido técnico) y, con esta finalidad, el mensaje suele incorporar una determinada cantidad de redundancia que garantiza la recepción correcta del mensaje.

El receptor que descodifica el mensaje utiliza un **código** común con el transmisor que lo ha codificado, y el resultado de esta descodificación es una traducción que se interpreta mediante el **código** común que tienen el emisor y el destinatario.

Por lo tanto, el transporte de información de emisor a receptor implica la presencia de un sistema constituido por los elementos que se acaban de representar. Este transporte es un proceso que debe superar dificultades (denominadas *interferencias* o *ruido*).

- La transmisión de información con éxito comporta la presencia de una serie de elementos que constituyen un sistema.
- El canal permite transportar señales de emisor a receptor.
- Cada mensaje está constituido por un conjunto de señales.
- Con el objetivo de recuperar la información contenida en el mensaje, el receptor tiene que compartir código con el emisor.

¿Qué es la información según Shannon y cómo se mide?

Hay que concretar el concepto de información y establecer y analizar su unidad mínima, el bit. Para la teoría de la comunicación (disciplina puntal en la informática a partir de los trabajos de Claude Shannon), la información no es lo que generalmente entendemos como tal, no tiene nada que ver con el sentido o significado de la información.

La información, según la teoría de Sannon, es la medida de la imprevisibilidad de un mensaje, es la medida de la libertad de elección que tiene el emisor o la fuente al seleccionar un mensaje.

La comunicación se empieza a producir en el caso más simple cuando el emisor puede escoger entre dos opciones (sí/no, abierto/cerrado, verdadero/falso, activo/inactivo, por ejemplo). Por este motivo, este caso más elemental recibe el nombre de ***binary digit*** (bit). Así pues, la información se medirá en bits.

Cantidad de información

Un mensaje sensato y uno estúpido pueden contener la misma cantidad de información en términos de teoría de comunicación.

Un bit es la cantidad de información contenida en un mensaje o éxito, cuya probabilidad de aparición es ½.

Con un solo bit, se puede hacer poca cosa a la hora de tratar la información de manera automática, sólo tenemos dos mensajes posibles, 0 y 1. Sin embargo, si tenemos dos bits uno al lado del otro ya podemos emitir cuatro mensajes: 00, 01, 10, 11.

Número de mensajes

Esto aumenta alarmantemente, porque con tres bits, tendremos ocho mensajes: 000, 100, 010, 001 ,110, 011, 101, 111.

Con cada bit nuevo añadido, se duplica la cantidad de mensajes posibles. Podemos obtener la cantidad de información aplicando la fórmula siguiente $2^i = Ms$, siendo i el número de bits utilizados y Ms el número de mensajes diferentes posibles.

Así pues, la información que incluye un mensaje tiene que ver con los mensajes que le son alternativos. Cuando el LED del monitor está encendido, recibimos el mensaje "Está activo", que contiene un bit de información, ya que sólo caben dos mensajes posibles.

Lo que hacen nuestros ordenadores es traspasar información de un lugar a otro. Cuando tecleamos en nuestro ordenador, le comunicamos una serie de mensajes, los archiva, los procesa y los envía de unos periféricos a otros, según el caso.

Para realizar estas tareas, lo ayudan una serie de programas que también son información, información que le comunica su disco duro en forma de instrucciones para ejecutar. Cuando se comunica con otros ordenadores, envía y recibe información.

Cuando se habla de automatizar el procesamiento de la información, no se refiere sólo a los cálculos que hace el ordenador cuando se le pida, sino a automatizar la desorganización que se produce en la información.

Por medio de la electrónica, todos y cada uno de los elementos de nuestro ordenador se comunican activamente con los otros y generan una serie de procesos comunicativos que requieren más o menos un tratamiento de los bits.

1.4.4. Los ordenadores tratan información

Hacemos un análisis a grandes rasgos de lo que comporta el tratamiento de información. Hoy día, la información tiene un papel determinante en muchos campos de la sociedad. Incluso decimos que "quien tiene información, tiene poder". Sin embargo, ¿qué entendemos por información? Si disponemos de un conjunto de datos, ¿equivaldría a decir que disponemos de información?

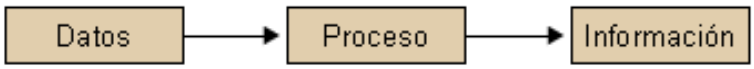
Elecciones generales

Supongamos el caso de unas elecciones generales: en el mismo momento en el que se cierran los colegios electorales, empiezan a salir los primeros resultados, sobre todo de encuestas realizadas en las salidas de los colegios. Cada una de estas encuestas no son nada más que datos, pero en conjunto pueden prever quién gobernará los próximos años, si se ha producido un voto de castigo a un determinado grupo parlamentario, etc.

Mismos datos, información distinta

También podemos ver en los medios de información que unos mismos datos, según los ojos con los que se miren, tendrán un valor u otro: parece que todo el mundo gana y que no hay perdedores. Por lo tanto, la información extraída a partir de los mismos datos no siempre es la misma.

Sin embargo, de lo que no tenemos ninguna duda es de que, para llegar a la información, hemos tenido que realizar un proceso en los datos. Esto nos lleva a plantear el gráfico siguiente:



Si lo miramos al revés, podemos afirmar que para llegar a la información necesitamos realizar un procesamiento en los datos. Los procedimientos, metodología, infraestructura, etc. utilizados para realizar este proceso de transformación de datos a información son lo que se denomina **sistema de información**, y cuando esta transformación de datos se hace con la ayuda de los ordenadores, entonces decimos que utilizamos un **sistema informático**. Ésta es la razón para que esta asignatura se denomine así.

Un sistema informático realiza, mediante ordenadores, el procesamiento de datos y, por lo tanto, facilita la obtención de información a partir de los datos.

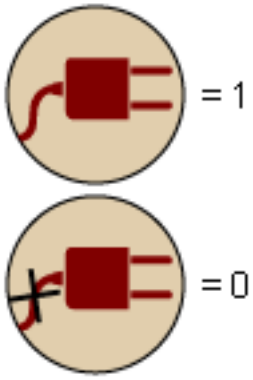
1.4.5. Comunicación con el hardware

Ahora aplicamos los conceptos de información y de comunicación en un caso concreto de la comunicación intermaquinaria e interpersonal. Podemos utilizar indistintamente las palabras **computador** y **ordenador**, dado que hacen referencia al mismo aparato. El motivo lo podemos encontrar en sus orígenes: en países de habla inglesa y en Alemania, Japón, etc. utilizaban el término *computador*, mientras que en Francia y España se utilizaba el término *ordenador*.

Llegado este punto, y de una manera muy genérica, podríamos decir que:

Un ordenador es un aparato digital que mediante un conjunto de pasos que hay que seguir (**programas**) permite obtener, procesar, almacenar y tratar los datos.

Si abrimos un ordenador, encontraremos muchos componentes electrónicos, cables, chips, etc. Por lo tanto, para hablar, le tendríamos que entregar señales eléctricas. Sin embargo, no nos interesa cuánto voltaje o tensión le llega, sino más bien si le llega tensión o no. Esto se debe al hecho de que los sistemas electrónicos digitales sólo pueden adoptar dos valores posibles: *on* y *off*. Es decir, *llega corriente* o *no llega corriente*.



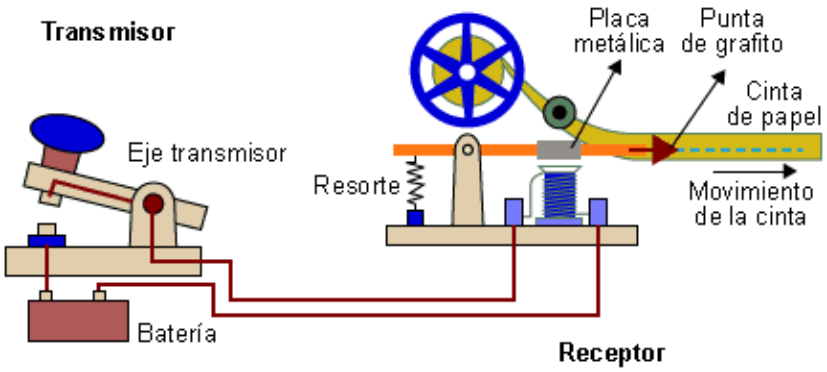
On/off

Por este motivo, decimos que un ordenador trabaja con lenguaje binario. Como ya hemos estudiado, la menor unidad del lenguaje binario es el bit, que puede significar el 0 y el 1.

Si conseguimos encontrar una relación entre el lenguaje natural y el lenguaje binario, habremos encontrado una manera de comunicarnos con el ordenador.

Queda claro que tenemos que buscar un sistema que sea cómodo para el usuario para trabajar con el ordenador, ya que ir encendiendo y apagando interruptores (que sería equivalente a poner 1 y 0) no parece la tarea más apropiada.

Esquema telégrafo



El sistema digital más antiguo que se conoce es la telegrafía. Hacia mediados de siglo XIX, Samuel Morse (1791-1872) inventó un sistema de codificación que permitía transformar letras del abecedario en impulsos eléctricos. Sólo había dos posibles valores y dependían del tiempo que se pulsara el botón del telégrafo: **punto** y **raya**.

El punto equivale a 1/25 s y la raya equivale a 3/25 s (es decir, la duración de tres puntos).

La codificación del alfabeto quedaba de la manera siguiente:

A · –	J · – – –	R · – ·
B – ···	K – · –	S ···
C – · – ·	L · – ··	T –
D – ··	M – –	U ·· –
E ·	N – ·	V ··· –
F ·· – ·	Ñ — · —	W · – –
G – – ·	O – – –	X – · · –
H ····	P · – – ·	Y – ·· – –
I ··	Q – – · –	Z – – ··

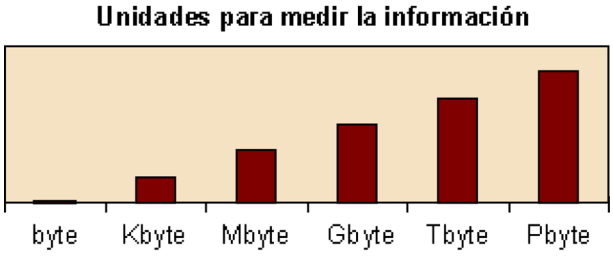
1 ·– – – –	6 – ····
2 ·· – – –	7 – –···
3 ··· – –	8 – – – ··
4 ···· –	9 – – – – ·
5 ·····	0 – – – – –

Mediante una combinación de puntos y rayas, se podían transmitir mensajes de una punta a la otra del país. Sin embargo, más que el invento, lo que nos interesa es el sistema de codificación. Hay una analogía muy grande entre esta codificación y la que podemos utilizar para comunicarnos con los ordenadores. En los dos sistemas, sólo hay dos valores posibles: punto-raya y 1-0.

Para hacer esta conversión no será necesario tener en la cabeza la tabla de conversión, como tenía el señor Morse, sino que el ordenador tiene mecanismos automáticos que nos harán esta conversión. Se trata de que el ordenador nos facilite la vida, no de que nos la complique.

En definitiva, para comunicarnos con el ordenador utilizaremos un sistema de codificación que nos permita pasar de un lenguaje que nosotros entendemos a otro que entiende el ordenador (el lenguaje binario).

De la misma manera que para hablar de distancia utilizamos las unidades de longitud o para hablar de líquidos utilizamos unidades de volumen, para hablar de información que almacenamos en la memoria necesitaremos unas unidades para medirla. Ya hemos visto que la menor unidad de información que se puede tener es el bit, pero para trabajar con cantidades de información mayores, necesitaremos unidades mayores.



Utilizaremos con mucha frecuencia estas unidades, no sólo para hablar de la capacidad de la memoria principal, sino para saber cuánto ocupa un documento de texto, una canción, un vídeo, qué espacio libre hay en un disquete, etc.

Para hacer cálculos

Si queréis practicar para hacer cálculos con bits, bytes, megabytes, etc., podéis ir al apéndice II.

1.4.6. Soportes físicos de información

La necesidad de mantener archivada la información como parte del tratamiento hace patente la necesidad de unos soportes físicos con discos, disquetes y unidades de almacenamiento de distintos tipos. Hay que hablar tanto de su estructura física y capacidad como de su lógica interna.

Ya hemos comentado anteriormente que los datos que puede haber en la memoria principal en un momento determinado son temporales, y es necesario el suministro eléctrico para no perderlos. Teniendo en cuenta esta afirmación, nos planteamos lo siguiente:

¿Dónde podemos almacenar un documento de manera que cuando volvamos a poner en marcha el ordenador lo volvamos a encontrar?

Queda claro, por lo tanto, que el objetivo principal de estos soportes físicos es garantizar que la información esté resguardada y, por lo tanto, tendrá mucha importancia en la elección de uno de estos dos factores: **capacidad de almacenamiento** y **velocidad**.

El almacenamiento se produce realizando una señal en el soporte por cada bit que queramos definir. Vista una unidad de espacio, la ausencia/presencia de señal determinará si se trata de un cero o de un uno, respectivamente.

Las señales se van haciendo una al lado de la otra en una fila. Cada una de estas señales la podemos denominar *celda*, si entendemos que cada celda guarda un bit de información único, en estado activo o inactivo (1, 0).

Las unidades de almacenamiento que tenemos crean la señal de tres maneras diferentes: **magnética** -en este caso, están fabricadas con una aleación de aluminio con un recubrimiento magnético-, **óptica** -hechas de aluminio reflector que se utiliza para reflejar los rayos láser- y **magnetoóptica** -que constan de dos capas, una formada por partículas magnéticas y una segunda de aluminio reflector, y que combinan la limpieza de lectura de los cabezales láser de los discos ópticos con la tradicional escritura por medio de cabezas magnéticas.

En el primer caso, se magnetiza una superficie y después se lee si la superficie transmite o no la electrificación, de cara a señalar un 1 o un 0; en el segundo caso, se perfora o calienta la superficie de manera que al reflejar o refractar sobre la superficie, o al no hacerlo, conseguimos el mismo resultado; en el tercer caso, se combinan las dos tecnologías, se magnetiza a la hora de crear la señal y se lee desde un dispositivo óptico.

A causa de la física de cada solución, las que magnetizan son óptimas para crear soportes de lectura-escritura, mientras que la óptica es más propia para una única escritura y lectura múltiple.

Los **discos ópticos** utilizan dos tecnologías para el almacenamiento de datos: WORM[9] y CD-ROM[10]. Los **discos magnetoópticos** utilizan la tecnología WM-RA[11], que permite leer y escribir tantas veces como sea necesario.

(9)WORM (*write once read many*) quiere decir 'escribe una vez, lee muchas'.

(10)CD-ROM (*compact disk read only memory*) quiere decir 'disco compacto de sólo lectura'.

(11)WMRA (*write many read always*) quiere decir 'escribe y lee muchas veces'.

Este aspecto también nos puede servir para clasificar las diferentes unidades de almacenamiento. Algunas son de una única escritura, de manera que, una vez establecida la información, no se puede cambiar, ya que la realización de la señal en el soporte no permite un regreso: es una modificación permanente y, en consecuencia, la definición del soporte es memoria de sólo lectura (ROM).

Discos ópticos

Los primeros discos ópticos aparecieron al final de la década de los ochenta. Los diferentes tipos de discos ópticos son laservisión o videodisco, CD (*compact disc digital audio*), CD-ROM (*read only memory*), CD-V (*compact disc video*), CD-ROM SHA (*extended architecture*), CD-I (*compact discinteractive*) y DV-I (*digital video interactive*).

Por otra parte, hay otros que son un dispositivo que permite escribir varias veces señales distintas en la misma área del soporte. De este modo, la información se puede modificar y cambiar por una diferente; en este caso, se trata de memorias de acceso aleatorio (RAM).

En este segundo grupo, encontraríamos aquéllas en las cuales la permanencia de la información depende de la fuente de alimentación (volátiles) y, en consecuencia, la pérdida de la alimentación produce la pérdida de la información. Éste sería el caso de los módulos de memoria DIMM[12] o SIMM[13] que tenemos en nuestro ordenador.

(12)DIMM (*dual in-line memory module*) quiere decir 'módulos de memoria en línea duales'.

(13)SIMM (*single in-line memory module*) quiere decir 'módulos de memoria en línea singulares'.

Otros soportes mantienen sus señales, aunque la fuente de alimentación se pierda. Es el caso, por ejemplo, de nuestro disco duro que, aunque permite que se escriba en el mismo muchas veces, guarda la información cada vez que apagamos y encendemos nuestro ordenador.

La información que puede caber en un soporte depende de la superficie aprovechable. La medida natural sería superficie/unidad de superficie = número de bits de capacidad, de manera que una superficie de diez centímetros cuadrados que necesitara un centímetro cuadrado para cada señal (10 / 1 = 10) podría almacenar diez bits.

Naturalmente, el almacenamiento por centímetro cuadrado suele ser mucho mayor, de manera que en un disquete convencional podemos archivar 1,44 MB, y en un disco duro del tamaño de una pastilla de turrón, hasta 120 GB, y pensemos que dentro del disco duro lo que soporta la información son cilindros; lo que vemos nosotros no es más que una carcasa.

Así pues, también podemos dividir las unidades de acuerdo con su capacidad. Hay de alta capacidad y de baja capacidad, de acuerdo con la cantidad de información que pueden almacenar.

Las unidades también pueden disponer de un formato lógico, que sería la organización previa que daríamos al espacio de almacenamiento del que disponemos.

Las particiones

Aunque hoy día no hay ningún problema para gestionar grandes unidades físicas en un PC, lo habitual es dividir la unidad física en unidades lógicas, que se denominan **particiones**, es decir, sería como poner vallas en un jardín y crear con esto diferentes espacios de la misma naturaleza pero en los cuales sabemos que no tenemos que transitar de uno a otro.

La gestión de estos espacios lógicos del disco se soluciona mediante información que se suele depositar y estructurar en el mismo disco. Esto define diferentes áreas como las siguientes.

- El **sector de arranque**: es el primer lugar que se lee de manera automática y contiene la información necesaria con el objetivo de poner en marcha la gestión de la información guardada en la unidad de almacenamiento.

- La **tabla de asignación de archivos (FAT)**: guarda información de los archivos, nombres, lugar, de inicio y de fin y otros datos necesarios para la gestión de archivos.

- **Directorio raíz**: es el lugar de partida de la estructura más próxima al usuario en la gestión de archivos. Cuelgan el resto de las carpetas y los archivos guardados. Es la lista de ficheros, vacía en principio, que contiene todo disco. Ocupa un número fijo de sectores, que determinan la capacidad de crear subdirectorios, y se encuentra detrás del último sector de la FAT.

- **Área de datos**: donde se depositarán, si todavía no se ha hecho, los archivos, las carpetas y las distintas informaciones de los usuarios de la unidad.

Los parámetros que se tienen que considerar en una unidad son tipos de disco, capacidad, tamaño, tiempo medio de acceso, velocidad de transferencia, velocidad de rotación, número de superficies, número de cabezas, número de pistas, número de sectores por pistas, número de palabras por sector, densidad máxima y código de grabación.

Históricamente, hay cinco tipos básicos de unidades: **discos de cabezas fijas**, con una cabeza individual de lectura/escritura para cada pista; **paquetes de discos**, que son distintos platos que giran conjuntamente en torno a un eje común, las cabezas de lectura/escritura son móviles y hay uno por superficie; **discos cartucho**, único plato con dos superficies de grabación; **discos duros**, disco de pequeño tamaño pero de gran precisión y con una gran capacidad de almacenamiento, las cabezas van más próximas a la superficie que en las unidades anteriores y se consigue, de esta manera, gran densidad de grabación; y **disquetes**, las unidades de *floppy disc* habituales.

¿Cuáles son las unidades de almacenamiento más corrientes?

- **Unidades de disquete**, también denominadas *floppy discs*, en inglés, o discos flexibles (del nombre en inglés viene la denominación FD), probable-

mente a causa de la ductilidad que les daba el material en el que estaban fabricados y su escaso grosor, o bien por el hecho de ser extraíbles (están compuestos por una lámina de poliéster -plástico flexible- de manera circular). La capacidad de almacenamiento es de 1,44 MB y en algunos formatos incluso el doble; hoy día casi ya no se utilizan.

Unidades de disquete

De hecho, es fácil encontrar hoy día ordenadores que ya se venden sin esta unidad, puesto que son capaces de arrancar desde el CD-ROM y el sentido de la disquetera no estaba vinculado a su capacidad, realmente obsoleta, sino al hecho de que la disquetera servía para iniciar los ordenadores cuando estaban vacíos o bien cuando había una caída del sistema irrecuperable.

- **Unidades de disco fijo o disco duro** (*hard disk*, 7.200 rpm[14]). Deben el nombre al hecho de que los cilindros sobre los cuales se escribe y se lee la información ya tienen una determinada rigidez y un mayor grosor, y son mayores la capacidad, la fiabilidad y la permanencia de la información. Suelen estar instalados en el ordenador cuando lo compramos, y el lugar donde se instalan es el sistema operativo de nuestro ordenador. Nada impide, por otra parte, añadir uno o varios discos nuevos a nuestro equipo si los que tenemos nos han quedado pequeños. Normalmente, se conectan al IDE[15] de la placa base, pero también es muy habitual que puedan funcionar por medio de una tarjeta SCSI[16] o bien algún tipo de conexión rápida, como después comentaremos.

[14]rpm: revoluciones por minuto.

[15]IDE: *integrated drive electronics*

[16]SCSI: *small computer system interface*

Unidades de disco fijo o disco duro

Hoy día, y en el mundo del PC de mesa, los discos llegan hasta 160 GB. La estructura física de los disquetes y de los discos duros es parecida. En los dos casos, podemos distinguir los lados que permiten escritura: las pistas -las *rasts* de datos que admite el soporte- y los sectores -las subdivisiones que se crean en las pistas. Suelen tener 512 bytes.

- **Unidades ZIP** (IOMEGA) o BACKUP de alta capacidad. La disquetera es externa y suele ser de un color azul oscuro y se puede conectar a IDE o SCSI, utiliza disquetes que tienen un tamaño y un volumen un poco superiores que un disquete de 3 1/2 o bien semejantes a una cinta de casete en otros casos, pero el más habitual es el disquete IOMEGA.
  Son adecuados para archivar todos los archivos referentes a un mismo proyecto si éste no contiene demasiada información. IOMEGA también tiene otro tipo de discos más parecidos a una unidad de disco duro separada entre disquetera y disquete, reciben el nombre de JAZ y almacenan hasta uno o dos GB.

Unidad ZIP

- **Unidades disco compacto (CD)**. Son los habituales CD que utilizamos hoy día y tienen una gran precisión. Pueden almacenar hasta 800 MB de información. Hay una única escritura (R) y escritura múltiple (RW). Los de una única escritura se pueden grabar varias veces, pero hasta que se llega al límite de su capacidad, dicho de otra manera, tienen un umbral de reutilización determinado, pero no se pueden reescribir indefinidamente. Los RW permiten la gestión del soporte como si se tratara de un disco duro o un disquete, se pueden reutilizar un número determinado de veces y, evidentemente, aprovechar en función de la calidad y la duración física.

Unidades de disco compacto

Sólo hay que recordar que tres minutos de grabación normal sin comprimir pueden llenar un CD-ROM de 800 MB.

- **Unidades de DVD**[17] que son capaces de almacenar hasta 4,7 Gb. También hay una única escritura (R) y escritura múltiple (RW). Su extensión se ha debido fundamentalmente al hecho de que son capaces de contener tanta información, que pueden incluir copias originales de los estudios de producción cinematográfica, de manera que lo que vemos en casa en un televisor de alta definición es como el original.
  Realmente, la gran capacidad del DVD sólo se puede aprovechar para almacenamientos de vídeo, gráficos o de audio en el entorno del aficionado.
  Se pueden encontrar DVD-ROM de *capa simple*, que pueden transferir unos 1,3 Mb por segundo, y almacenar en el mismo espacio que un CD-ROM 4,7 Gb, tal y como hemos señalado anteriormente, y también DVD-ROM de *doble capa*; en una, semitransparente reflectiva con oro, se pueden almacenar 3,8 Gb, y esta capa se encuentra debajo de la otra capa reflectora -con los habituales 4,7 GB-, metalizada con plata. En total, 8,5 Gb.
  Finalmente, está el DVD-RAM (RW), que tiene limitaciones técnicas que le impiden almacenar más de 2,6 Gb.

(17)DVD: *digital versatil disk*

DVD

- Hoy día, también se han extendido mucho las **unidades removibles**, que funcionan por medio del sistema de transmisión universal **USB**, incluso

FireWire o Ilink. En la vertiente de USB tenemos lápiz de memoria, *pen drive* o *flash memory*, que parecen pequeños bolígrafos o mecheros que se pueden llevar en el bolsillo y permiten almacenar 128 Mb o 512 Mb, o incluso más, aunque ya en este caso quizá será mejor otro tipo de unidad. Su ventaja es la facilidad de uso, que permite cargar una cantidad de información importante en muy poco espacio y con muy poco peso.

Lápiz de memoria

Los periféricos lápices de memoria permiten almacenar información de una manera mucho más segura que los disquetes, ya que no hay ningún tipo de fricción. Dado su tamaño (unos 8 × 2 cm), son mucho más transportables.

- Otro tipo de unidad, también relacionado con el tratamiento de vídeo, son los **discos de gran capacidad** y conexión **USB**[18] o **FireWire**. Estas unidades tienen una cantidad interminable de capacidad, ni más ni menos que 120 Gb. Su uso en la edición de vídeo doméstico se está haciendo imprescindible a causa de la gran cantidad de almacenamiento que requieren las películas de vídeo.

Conexión USB

Los discos duros con conexión USB son muy utilizados por los informáticos para hacer copias de imagen de los discos duros donde se instalan todos los programas y los datos del ordenador. Si por desgracia nos entra un virus y estropea nuestra información, la podremos restaurar de estas copias de imagen sin tener que volver a instalar nada más. Podríamos decir que nos deja el ordenador tal y como lo teníamos en el momento de hacer la copia de imagen.

- Los parámetros clave de las unidades de almacenamiento son velocidad de acceso y capacidad.
- Las unidades de almacenamiento más corrientes son: disquete, disco duro, ZIP, CD y DVD, removibles y de gran capacidad.

(18)USB: *universal serial bus*

Cámaras de fotografía

Las cámaras de fotografía y digitales pueden, hoy día, bajar grandes cantidades de información en nuestro ordenador y una unidad de este tipo nos puede facilitar el almacenamiento y la organización.

1.4.7. Álgebra de Boole

Hasta ahora hemos estado hablando extensamente de cómo podemos codificar los datos, pero no hemos hecho ningún comentario de cómo se tratan estos datos.

El álgebra de Boole es un sistema matemático idóneo para modelizar los circuitos electrónicos del ordenador, dado que se basa en la teoría de que sólo hay dos estados posibles, dos valores posibles que en nuestro caso serán 1 y 0. Es el sistema que permitirá imitar el pensamiento humano dentro de un hardware organizado.

George Boole

El señor George Boole (1815-1864) investigó las leyes fundamentales que rigen las operaciones que llevan a los humanos a alcanzar razonamientos. Para hacerlo, creó el álgebra que hoy día lleva su nombre.

Las operaciones lógicas que veremos a continuación determinan este sistema matemático. Un siglo después de ser diseñado, éstas se utilizarían para implementar los circuitos lógicos o circuitos electrónicos, según si los entendemos desde el punto de vista matemático o desde el punto de vista eléctrico, pero que se alimentan de uno de los dos posibles estados: 1 o 0, cierto o falso, corriente o no corriente.

A pesar de esta posibilidad de denominar a estos posibles estados de maneras diferentes según el contexto en el que nos encontramos, y para facilitar la lectura de los próximos apartados, nos centraremos en la expresión numérica 1 s y 0 s.

Cada uno de los circuitos lógicos que realizan una operación lógica se denomina **puerta lógica**. Estas puertas están compuestas, a su vez, por *transistores* mediante los cuales, junto con el aprovechamiento de los *semiconductores*, se consigue implementar de manera física la teoría que en su día nos dejó el señor Boole. Encadenando un grupo de puertas lógicas, podemos llegar a construir operaciones más complicadas.

Circuitos integrados

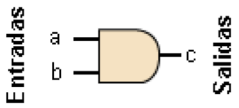
De hecho, sería muy parecido al juego de niños Lego. Podemos construir pequeños objetos que, al mismo tiempo, podemos juntar para hacer otro mayor. Es lo que denominamos *circuitos integrados*, que son el paso previo a los procesadores.

El álgebra de Boole es un sistema matemático idóneo para modelizar los circuitos electrónicos del ordenador.

1.4.8. Puertas lógicas

Derivadas de la lógica de enunciados, son las que permiten que la lógica del pensamiento sea aplicada por un hardware sobre la información y extraer de este modo reacciones "inteligentes" o, al menos, decisiones precisas que llevan un alto proceso de evaluación. Las puertas lógicas más simples están formadas por dos entradas y una salida. Sería como si entraran dos hilos eléctricos, se realiza la operación y da una salida para un solo hilo.

Ejemplo de puerta lógica



Según los valores que puedan tener las entradas de una puerta lógica y de la operación matemática que realice, tendrá una salida con diferentes valores resultantes. Las tablas que muestran todos estos valores de entradas y salidas posibles se denominan **tablas de verdad**.

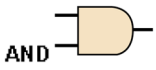
Cada uno de los hilos de entrada de una puerta lógica los denominaremos *variables*, y el número de posibles valores de salida o, dicho de otra manera, el número de posibles combinaciones de entrada será equivalente a $2^{\text{número de variables de entrada}}$.

Así pues, en el ejemplo anterior, de dos entradas tendremos cuatro posibles salidas ($2^{\text{2 entradas}} = 4$ salidas).

Entradas		**Salidas**
a	**b**	**c**
0	0	**?**
0	1	**?**
1	0	**?**
1	1	**?**

Para facilitar la lectura de las tablas de verdad y de manera estándar, siempre escribiremos las diferentes combinaciones de las entradas como si se tratara de un sistema de numeración binario (00, 01, 10, 11).

Empecemos a hablar de las puertas lógicas más sencillas: la puerta AND, la puerta OR y la puerta NOT:



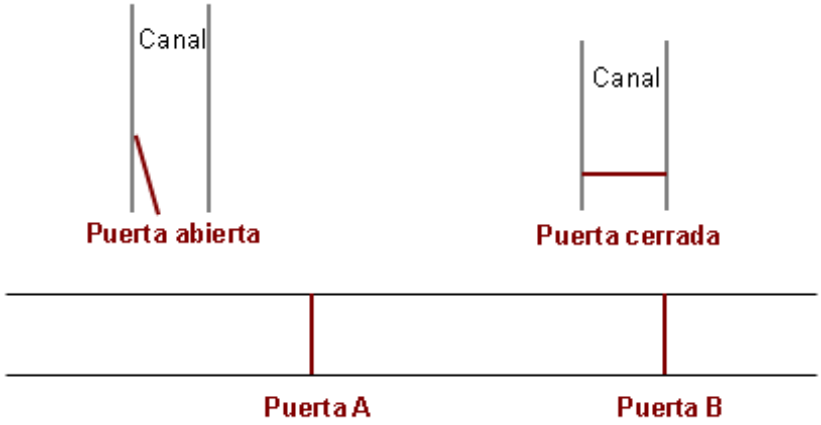
Entradas		**Salidas**
a	**b**	**c**
0	0	**0**
0	1	**0**
1	0	**0**

Entradas		Salidas
a	b	c
1	1	**1**

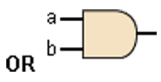
La operación AND retorna un 1 sólo cuando todas sus entradas son un 1. El resto, retorna un 0.

Los canales de agua

Si intentamos buscar una analogía de la vida real que nos describa el funcionamiento de esta función lógica, podríamos encontrar, por ejemplo, el funcionamiento de los canales de agua para regar un campo de un campesino. Tenemos un canal de agua muy largo con dos puertas separadas entre sí que nos permitirán dejar pasar el agua o no. Si la puerta no deja que continúe circulando agua, diremos que es cerrada y, si deja circular el agua, diremos que es abierta.



Canales de entrada		Canal de salida
Puerta a	Puerta b	Campo
Cerrada	Cerrada	**No se riega**
Cerrada	Abierta	**No se riega**
Abierta	Cerrada	**No se riega**
Abierta	Abierta	**Se riega**



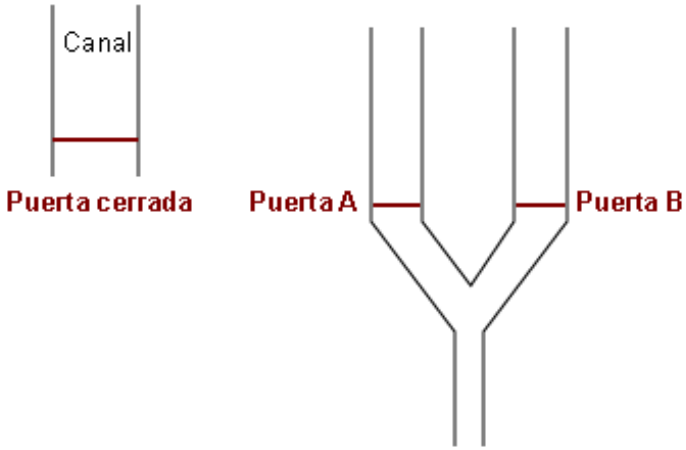
Entradas		Salidas
a	b	c
0	0	**0**
0	1	**1**
1	0	**1**

Entradas		Salidas
a	b	c
1	1	**1**

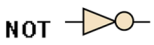
La operación OR retorna un 1 cuando una de las entradas sea un 1. El resto, reanima un 0.

Los canales de agua

Seguimos con el ejemplo de los canales, con el objetivo de regar un campo de un campesino, pero ahora tenemos dos canales de agua independientes que van a parar a un canal mayor. Antes de llegar, encontraremos una puerta que no deja que continúe circulando agua (diremos que es cerrada) o que deja circular el agua (diremos que es abierta).



Canales de entrada		Canal de salida
Puerta a	Puerta b	Campo
Cerrada	Cerrada	**No se riega**
Cerrada	Abierta	**Se riega**
Abierta	Cerrada	**Se riega**
Abierta	Abierta	**Se riega**



Entradas	Salidas
a	c
0	**1**
1	**0**
0	**1**

Entradas	Salidas
a	c
1	**0**

La operación NOT retorna el valor contrario de la entrada.

Los canales de agua

En este caso, es difícil encontrar una analogía en el ejemplo de los canales para regar un campo. Para no intentar buscar otro ejemplo y con grandes dosis de imaginación, pensaremos que el campesino está enfermo y el hijo se encargará de abrir y cerrar la puerta. Sin embargo, el hijo está muy enfadado con su padre, porque para hacer esta tarea no podrá ir a jugar con sus amigos, y ha decidido hacer lo contrario de lo que éste le diga.

Cuando el campesino dice "Abre la puerta", el hijo la cierra y cuando el campesino dice "Cierra la puerta", entonces el hijo la deja abierta.

Imaginemos que con este ejemplo tenemos a una familia que no tomará ningún fruto del campo.

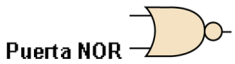
Como ya hemos comentado en un apartado anterior, Claude Shanon participó en la **teoría de la información y la comunicación**. Después de doctorarse en el Instituto de Tecnología de Massachusetts (MIT), estableció unos vínculos entre las matemáticas y la electrónica. Mediante la electrónica, pudo reproducir físicamente las diferentes teorías desarrolladas por George Boole.

Enlaces de interés

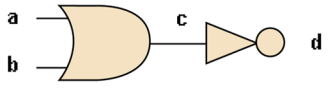
Instituto de Tecnología de Massachusetts (MIT).

Biografía de Claude Shannon.

Si juntamos algunas de estas funciones lógicas, encontramos como resultado otras nuevas. Por ejemplo, si juntamos una puerta OR con una puerta NOT, tenemos como resultado una puerta NOR.



Si hasta ahora hemos dicho que la puerta OR tiene como resultado un 1 siempre que un valor de su entrada sea un 1, ahora, que tenemos que negar lo que salga de esta puerta; diremos que la puerta NOR será 1 cuando sus dos entradas sean 0.

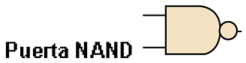


La tabla de verdad de la puerta NOR es la siguiente:

Entradas		Salidas	
a	b	c	d
0	0	**0**	**1**
0	1	**1**	**0**

Entradas		Salidas	
a	b	c	d
1	0	**1**	**0**
1	1	**1**	**0**

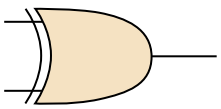
De la misma manera, tenemos la puerta AND, formada por la unión de la puerta AND y la puerta NOT.



Esta función tomará el valor de salida 1 mientras sus valores de entrada sean diferentes a 1, aunque también podríamos hacer esta otra lectura: se consigue el valor 0 cuando las entradas son un 1.

La tabla de verdad de la puerta NAND es la siguiente:

Entradas		Salidas
a	b	c
0	0	**1**
0	1	**1**
1	0	**1**
1	1	**0**



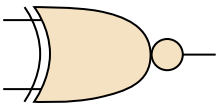
Puerta XOR
(también llamada OR-exclusiva)

Esta función tomará el valor de salida 1 cuando una y sólo una de sus entradas sea 1.

La tabla de verdad de la puerta XOR es la siguiente:

Entradas		Salidas
a	b	c
0	0	**0**
0	1	**1**
1	0	**1**

Entradas		Salidas
a	b	c
1	1	**0**



Puerta NOR EXCLUSIVA (NXOR)

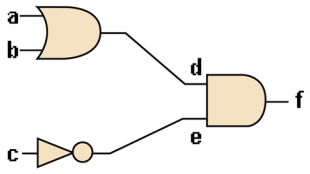
Esta función tomará el valor de salida 1 cuando sus entradas sean 1 o sean cero.

La tabla de verdad de la puerta NXOR es la siguiente:

Entradas		Salidas
a	b	c
0	0	**1**
0	1	**0**
1	0	**0**
1	1	**1**

Concatenación de puertas lógicas

Hasta ahora hemos comentado las puertas lógicas más básicas, pero veremos un ejemplo de cómo podemos conseguir encontrar la tabla de verdad de un circuito formado por distintas puertas lógicas.



A partir de las tablas de verdad, se consigue una función algebraica que la representa. Mediante un conjunto de teoremas y axiomas (de los cuales no hablaremos en esta asignatura), se consigue optimizarlas. Después, si se hace el proceso inverso, es decir, pasando de la función optimizada a puertas lógicas, se ha conseguido la mayoría de las veces reducir el número de puertas lógicas sin cambiar en ningún momento el valor de las salidas. Podríamos decir que se trata de circuitos de puertas lógicas equivalentes, pero simplificados.

Partimos de un circuito de tres entradas (a, b y c). Por lo tanto, la tabla de verdad tendrá ocho combinaciones posibles ($2^{3\ \text{entradas}}$). La tabla de verdad correspondiente sería la siguiente:

Entradas			**Salidas**		
a	**b**	**c**	**d**	**e**	**f**
0	0	0	**0**	**1**	**0**
0	0	1	**0**	**0**	**0**
0	1	0	**1**	**1**	**1**
0	1	1	**1**	**0**	**0**
1	0	0	**1**	**1**	**1**
1	0	1	**1**	**0**	**0**
1	1	0	**1**	**1**	**1**
1	1	1	**1**	**0**	**0**

El procedimiento que hemos seguido ha sido éste:

1) Conseguimos la tabla de verdad del OR (entradas a, b y salida d).

2) Obtenemos la tabla de verdad de la puerta NOT (entrada c y salida e).

3) Para acabar, realizamos la tabla de verdad de la puerta AND (entradas d, e y salida f).

De hecho, el sistema es el mismo, pero se trata de ir dividiendo el análisis desde el inicio y llegando al final.

Las combinaciones de puertas lógicas permiten realizar operaciones lógicas complejas sobre el hardware.

1.4.9. Software, hardware, instrucciones, datos y lenguajes de programación

Llegados a este punto, y antes de empezar a explicar sistemas operativos, debemos diferenciar entre distintos elementos con el objetivo de evitar confusiones terminológicas. Lo que hemos visto hasta ahora es que el ordenador trabaja con dos tipos de informaciones: **datos** e **instrucciones**. Gracias a estas instrucciones, el ordenador sabe qué debe hacer con los datos.

El conjunto de instrucciones que nos llevan a realizar una función determinada lo denominaremos **programa**. Ya hablaremos de esto con más detalle en capítulos posteriores. De momento, damos la idea para poder relacionar todos estos conceptos.

El **software** hace referencia al conjunto de programas que se pueden ejecutar en un ordenador. Y el **hardware**, a la parte física del ordenador: periféricos, CPU, memoria, etc.

Para hacer los programas, necesitamos un lenguaje que nos permita escribirlo cómodamente pero que, a la vez, mediante un proceso automático, lo pueda entender el ordenador. Como ya hemos comentado anteriormente, complicaríamos la vida del programador si tuviera que escribir los programas con ceros y unos. Denominaremos **lenguajes de programación** a los diferentes lenguajes formados por un conjunto de reglas y símbolos que permiten crear los distintos programas.

Un ordenador está compuesto por software y hardware.

2. Sistemas operativos

2.1. Introducción a los sistemas operativos: ¿por qué lo tenemos que hacer nosotros?

Hasta ahora, hemos visto que los ordenadores contienen sistemas muy complejos y que para hacerlos funcionar tendríamos que saber hablar con las impresoras, con el lector de CD-ROM, con los programas, etc.

Queda claro que esto no lo puede hacer el usuario y, por lo tanto, necesitaremos que alguien haga estas tareas de manera automática. El encargado es el denominado *sistema operativo*.

Cada día que pasa, los sistemas operativos son más amigables y transparentes para el usuario y facilitan más tareas de uso y, sobre todo, de mantenimiento del ordenador.

Un ordenador no puede funcionar si no tiene un sistema operativo.

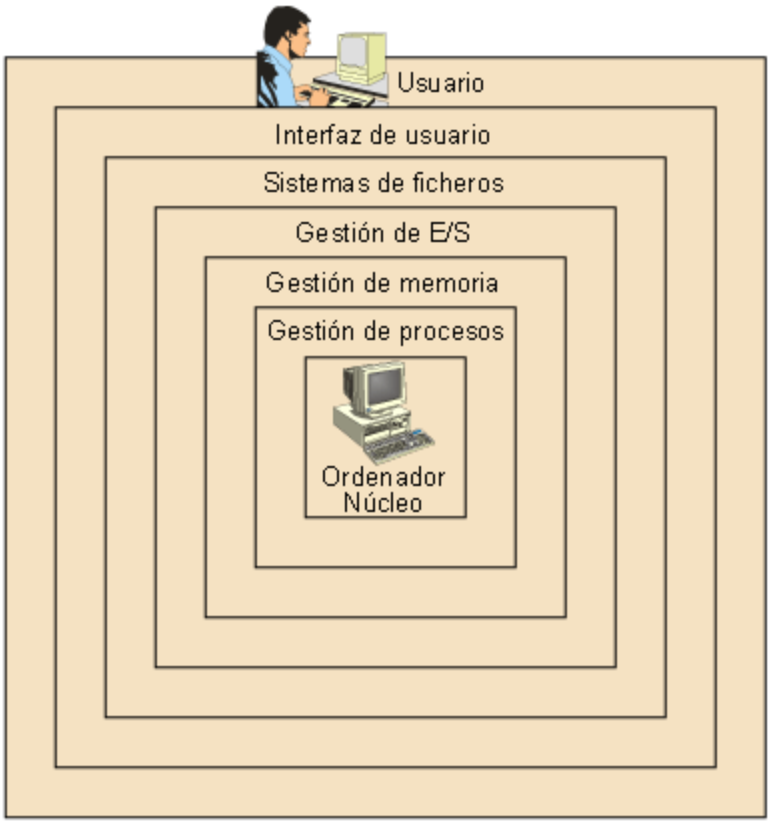
Ordenadores con dos sistemas operativos

Los ordenadores que tienen dos sistemas operativos instalados en diversas particiones, al ponerse en marcha, piden al usuario que indiquen con cuál quieren trabajar.

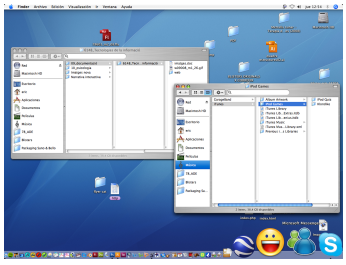
Para definir un *sistema operativo*, podríamos decir que es el lenguaje que utilizamos para comunicarnos con el ordenador. Hace de intermediario entre el usuario y las diferentes partes del ordenador (memoria, periféricos y CPU).

2.2. Partes de un sistema operativo

A continuación se relacionan las distintas partes de un sistema operativo, la función de cada uno y la clasificación de los sistemas según diferentes criterios. Hemos dicho que el sistema operativo es el lenguaje que utilizamos para comunicarnos con el ordenador, que de alguna manera hace de intermediario entre el usuario y las diferentes partes del ordenador. Entre el usuario y el ordenador, hay un conjunto de capas distribuidas jerárquicamente en el cual las más externas se apoyan con las más internas para realizar sus funciones. Si sumamos el conjunto de cada una de las funciones siguientes de las diferentes capas, obtenemos lo que hemos denominado *sistema operativo*.



¿Cuál es la función asignada a cada una de estas capas?



- **Interfaz de usuario**: es la capa que interactúa directamente con el usuario, y por lo tanto, se trata de hacer que el usuario se sienta cómodo interaccionando. Por ejemplo, es mucho más fácil e intuitivo que, para imprimir un documento, el usuario sólo tenga que arrastrar un documento hacia el dibujo de una impresora, y no que deba escribir una orden con una sintaxis determinada e indicar dónde está el documento, por ejemplo.

- **Sistema de ficheros**: se encarga de gestionar el sistema de ficheros del sistema operativo. Cuando se da una orden desde la interfaz de usuario para guardar un documento, por ejemplo, esta capa es la que determina cómo y dónde se debe guardar.

- **Gestión de E/S**: se encarga de regular el acceso al procesador de los diferentes periféricos y el flujo de entrada y salida desde cada uno de ellos.

- **Gestión de memoria**: se encarga de gestionar la memoria principal. Cuando un proceso necesita memoria, por ejemplo para guardar un dato, hace uso de esta capa.

- **Gestión de procesos**: se encarga de la gestión de los procesos para poder ejecutar los programas.

- **Núcleo**: es la parte más baja de un sistema operativo y, por lo tanto, la que habla más directamente el lenguaje que entiende el procesador de la máquina.

Entre el usuario y el ordenador hay seis capas del sistema operativo, que se apoyan jerárquicamente la una a la otra.

2.3. Clasificación de sistemas operativos

Según cuál sea el punto de vista con el que miramos un sistema operativo, lo podremos clasificar de un tipo o de otro. A veces, notamos que un mismo sistema operativo puede tener diferentes clasificaciones.

Por lo tanto, sería interesante hablar un poco de los diferentes tipos de clasificaciones de sistemas operativos:

- Sistemas operativos por su estructura.
- Sistemas operativos por los servicios que ofrecen.
- Sistemas operativos por la manera en la que ofrecen sus servicios.

De esta manera, podemos decir que un sistema operativo cualquiera puede ser de un tipo determinado según si tratamos la clasificación por su estructura, por los servicios que ofrece y/o por la manera en la que nos ofrece sus servicios.

2.3.1. Sistemas operativos por su estructura

Con respecto a la estructura, los sistemas operativos se clasifican en:

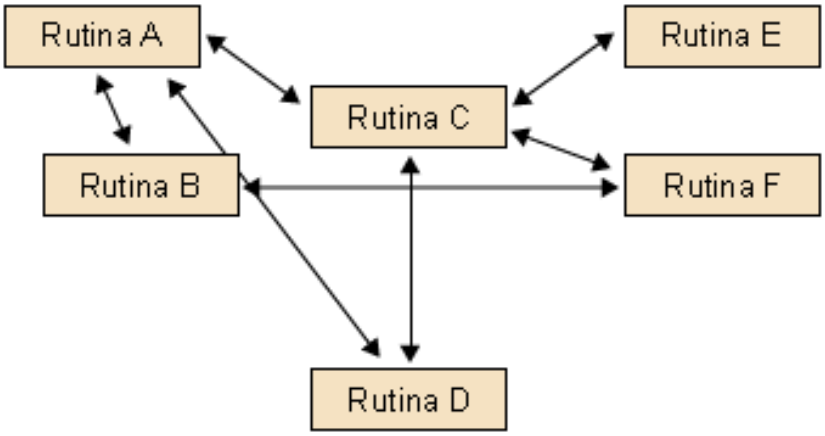
- Estructura monolítica.
- Estructura jerárquica.
- Máquina virtual.
- Cliente servidor.

Estructura monolítica

Los primeros sistemas operativos sólo podían ejecutar un único programa que, a la vez, estaba dividido en diferentes partes o rutinas entrelazadas de manera que se pudieran comunicar o llamar.

Eran sistemas operativos muy rápidos pero a la vez muy vulnerables, ya que el tratamiento que se hacía para protegerlos y los privilegios que se tenían que aplicar para utilizar determinadas rutinas que trataban directamente partes significativas de los recursos del ordenador eran nulos.

Ejemplo de estructura monolítica

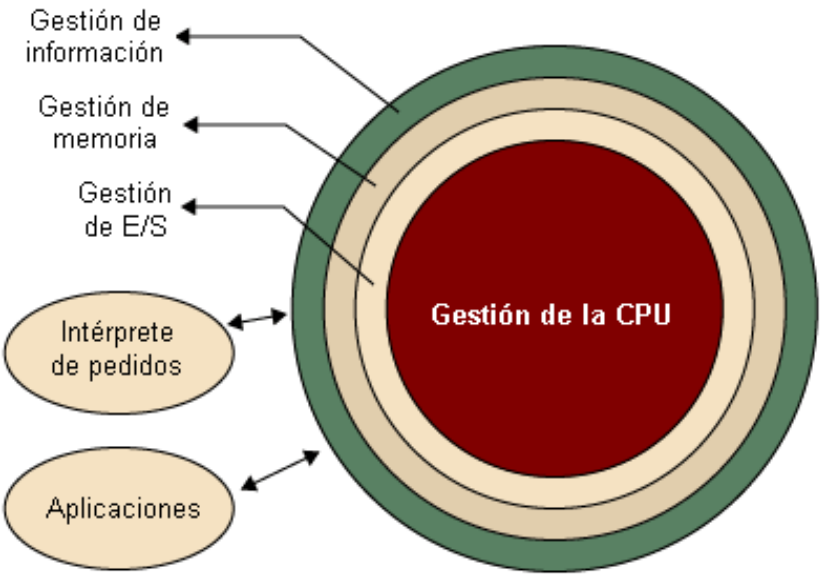


Estructura jerárquica

Estos sistemas operativos estaban divididos en diferentes capas concéntricas, en una jerarquía de niveles. Cada una de estas capas tenía una funcionalidad concreta, muy bien definida, y había un comunicador o interfaz para que se trataran entre sí.

El primer sistema operativo construido de esta manera fue el sistema THE, que fue fabricado en el Technische Hogeschool Eindhoven de Holanda por E. W. Dijkstra (1968) y sus alumnos.

Ejemplo de sistema jerárquico



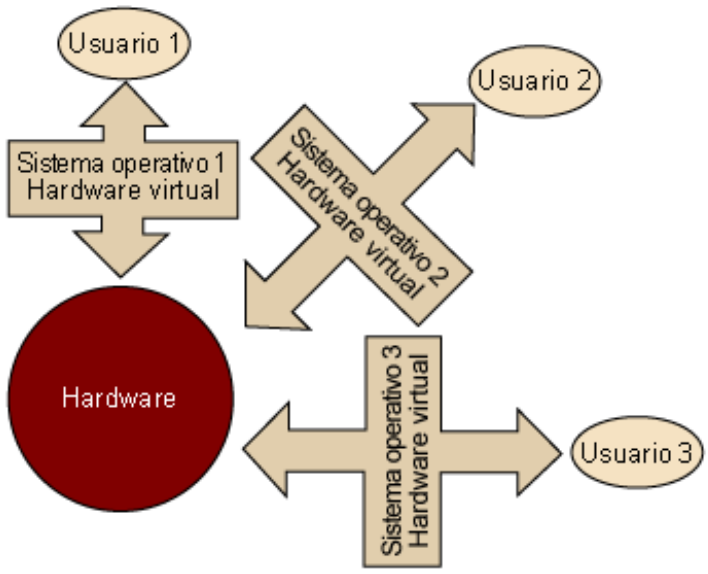
Con este modelo, se conseguía proteger los accesos no autorizados de capas superiores a capas inferiores y se aumentaba la seguridad del sistema operativo.

Máquina virtual

Se trata de un tipo de sistema operativo que permite hacer réplicas virtuales del hardware y, por lo tanto, permite que se puedan ejecutar diferentes sistemas operativos en la misma máquina.

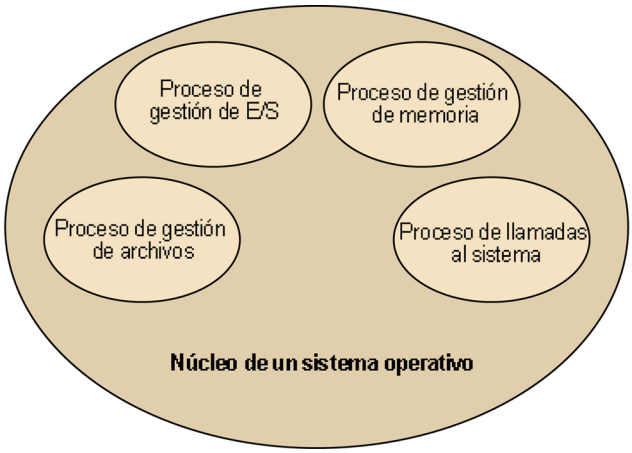
De esta manera, por ejemplo, podemos conseguir que en la misma máquina haya un usuario que ejecute un programa bajo un sistema operativo cualquiera y otro usuario que ejecute otro programa bajo un sistema operativo completamente diferente al anterior.

Ejemplo de máquina virtual



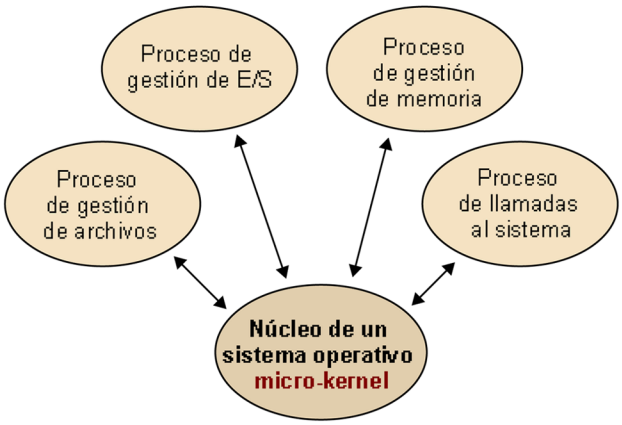
Cliente-servidor (*micro-kernel*)

Hasta ahora, los sistemas operativos sólo tienen un núcleo único que sólo se ejecuta en una única máquina.



Se trata de liberar del núcleo parte del trabajo que hacía normalmente. Se habla de los procesos servidor (los que ofrecen el servicio) y de los procesos cliente (los que utilizan los servicios ofrecidos).

De esta manera, el micronúcleo, *micro-kernel*, sólo tiene en cuenta un número limitado de operaciones y el resto (operaciones de gestión de memoria, de sistemas de archivo, de entrada/salida, de llamadas al sistema, etc.) se ejecutan por medio de procesos servidor que, a la vez, se pueden ejecutar en máquinas diferentes con arquitecturas distintas.



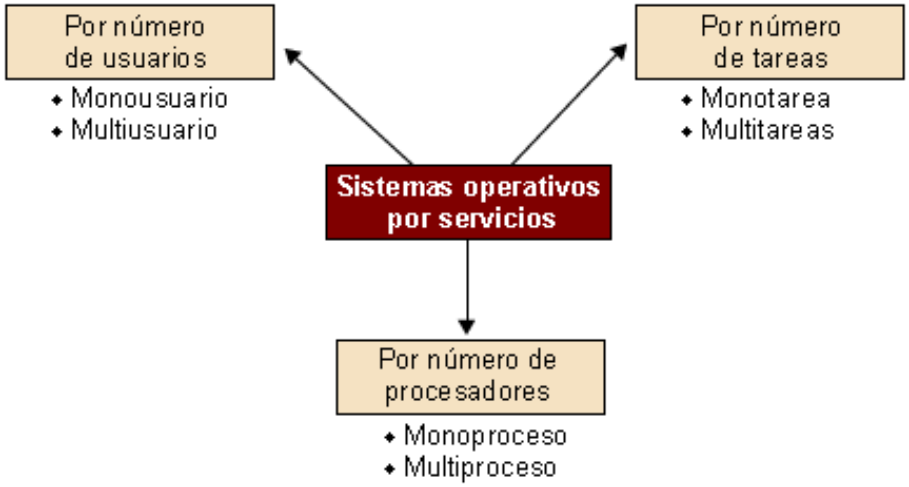
De esta manera, el sistema operativo se encarga principalmente de gestionar las comunicaciones entre los procesos servidor y los procesos cliente.

Se pueden ejecutar en todo tipo de máquinas, independientemente del tamaño. Algunos ejemplos de sistemas operativos del tipo micronúcleo son AIX, Minix, BeOS, AmoebaOS, etc.

Los tipos de sistema operativo según la estructura son monolíticos, jerárquicos, máquina virtual y cliente-servidor.

2.3.2. Sistemas operativos por los servicios que ofrecen

Ésta es la clasificación que más utiliza el usuario de ordenadores de la calle. Según si hablamos del número de usuarios, diremos que puede ser monousuario o multiusuario; si hablamos del número de tareas que realiza, entonces diremos que puede ser monotarea o multitarea; y si hablamos del número de procesadores, entonces hablaremos del monoproceso o multiproceso. Esta clasificación no es excluyente, es decir, podemos encontrar, por ejemplo, un sistema operativo que sea monousuario y multitarea a la vez.



Por el número de usuarios

- **Monousuario**. Estos sistemas operativos sólo soportan a un solo usuario a la vez. Este usuario puede estar ejecutando uno o más programas a la vez, pero hasta que no acabe podrá venir otro usuario a hacer su trabajo. Antiguamente, a los ordenadores personales se les atribuía este modelo de sistema operativo.

Ejemplo de monousuario

Ejemplos de este modelo serían MS-DOS, Windows 95, Windows 98 y MacOS.

- **Multiusuario**. Estos sistemas operativos permiten que trabajen a la vez distintos usuarios mediante algún sistema de comunicaciones (diferentes terminales conectados al ordenador, o conexiones remotas mediante alguna red de ordenadores).

Ejemplo multiusuario

Ejemplos de este modelo serían Windows 2000, Unix y Linux, Windows XP y Windows Vista.

Por el número de tareas

- **Monotarea o monoárea**. Son los que permiten ejecutar en el ordenador una única tarea por usuario en un momento concreto. Lógicamente, una vez acabada su tarea, este mismo usuario puede ejecutar otras, pero eso sí, de una en una.

Sistemas operativos monotarea y multiusuario

Un sistema operativo monotarea y multiusuario permitiría que diferentes usuarios pudieran ejecutar un único proceso en un momento concreto.

- **Multitarea o multiárea**. Se trata de sistemas operativos que permiten hacer distintas tareas o ejecutar diferentes programas a la vez en un momento concreto. Un ejemplo sería que un usuario, mientras recibe sus mensa-

jes de correo electrónico, pueda estar escribiendo un documento con un programa procesador de textos.

Por el número de CPU

Hay ordenadores a los cuales la arquitectura les permite disponer de más de un procesador.

- **Monoproceso**. Son los sistemas operativos que sólo pueden trabajar con una sola CPU.

- **Multiproceso**. Son los sistemas operativos que permiten trabajar con más de un procesador. Se trata de sistemas operativos muy complejos, ya que hay un añadido o una carga suplementaria con respecto a los sistemas operativos monoproceso: se debe gestionar el trabajo que tiene que hacer cada procesador de manera eficiente, para que todo procesador tenga un trabajo asignado en todo momento y así podamos sacar el máximo provecho.

Los tipos de sistema operativo según los servicios que ofrecen son monousuario/multiusuario, monotarea/multitarea.

2.3.3. Sistemas operativos por la manera en la que ofrecen sus servicios

Esta clasificación hace referencia a la manera en la cual el usuario puede acceder a los servicios que ofrecen los sistemas operativos mediante un medio de transporte (ordenadores interconectados).

Sistemas operativos de red

Se trata de sistemas operativos que permiten hacer determinadas tareas con ordenadores de otras máquinas utilizando un medio de transporte y un sistema cliente-servidor. Es el usuario el que tiene que conocer la manera de acceder, no se hace de manera automática. Algunos ejemplos de estas tareas que se permiten realizar serían los siguientes.

- Intercambiar información: transmisión de ficheros.

- Ejecutar órdenes en el ordenador remoto (al que estamos accediendo).

- Compartir recursos; por ejemplo, imprimir documentos en una impresora instalada en el ordenador remoto.

- Etc.

Sistemas operativos distribuidos

Se trata de un sistema operativo como el anterior (sistema operativo de red), pero el usuario no tiene que intervenir tanto, no necesita conocer la manera de acceder al mismo. Es como si se tratara de una única máquina u ordenador. Por ejemplo, para poder imprimir un documento en un ordenador remoto, si disponemos de un sistema operativo distribuido, no sería necesario que creara un vínculo con aquella impresora, sino que automáticamente ya nos aparecería.

La dificultad de estos sistemas operativos es muy grande, ya que pensamos que para ejecutar un programa en concreto se tiene que analizar la posibilidad de poder fragmentarlo de manera que diferentes CPU de máquinas distintas puedan realizar cada una su parte y después poder juntarlas. Y todo esto, intentando mejorar el tiempo con respecto a otros sistemas operativos que lo ejecutan en una sola CPU.

> Pongamos un ejemplo más real para ver esta dificultad de trabajar con distintos procesadores de diferentes máquinas: no es lo mismo que una sola persona cocine una tortilla de patatas, que deba cocinar mil tortillas de patatas. Necesitaremos a más personas (CPU) que nos ayuden: unas pelarán las patatas, mientras que otras calentarán el aceite de las sartenes y otras batirán los huevos. Todo el mundo lo hará desde su casa (máquinas). Deberemos estar muy atentos a que cada sartén tenga todo lo que necesita en cada momento (medio de comunicación entre máquinas).

Con este ejemplo queda clara esta dificultad, pero en sistemas operativos distribuidos la complejidad es mucho mayor y a veces querer fraccionar las tareas que se deben hacer implica aumentar el control de manera exagerada.

Más adelante, hablaremos un poco más sobre esto.

Los tipos de sistema operativo según la manera de ofrecer los servicios: en red, distribuidos.

2.4. Evolución de los sistemas operativos

A continuación, trataremos las distintas generaciones de hardware asociadas a los distintos hardware y sistemas que ha habido. Para poder explicar la evolución de los sistemas operativos, debemos hablar de la evolución de las arquitecturas de los ordenadores, ya que siempre han estado muy relacionadas. A medida que la tecnología ha ido evolucionando, los sistemas operativos también lo han hecho.

2.4.1. Primera generación (1946-1955)

El componente electrónico utilizado en la construcción de los ordenadores era la **válvula de vacío**. Estas válvulas funcionaban calentando una pequeña platina interior y controlando el flujo de electrones. Los ordenadores de esta generación eran muy grandes y pesados, y estaban formados por centenares de válvulas de vacío, que provocaba que se calentaran mucho.

Se utilizaban estos ordenadores para hacer cálculos matemáticos con finalidades militares (trayectorias balísticas).

Propiamente, estos ordenadores no disponían de sistema operativo, ya que los programas que se ejecutaban se cargaban directamente a la memoria de los ordenadores y los resultados quedaban grabados en tarjetas perforadas. Esto hacía que el trabajo fuera muy pesado. Por cada ejecución de una instrucción, se tenía que ir cambiando el cableado del ordenador.

ENIAC: *electronic numerical integrator and computer* (1945)

Peso aproximado: 30 toneladas
Ocupaba 150 m^2
Potencia de 150 kW

2.4.2. Segunda generación (1955-1964)

Apareció un nuevo componente electrónico utilizado en la construcción de los ordenadores, el **transistor**, desarrollado en el año 1947 por John Bardeen, Walter Bratain y Willian Shockley, ingenieros de los laboratorios Bell. Se sustituyeron las válvulas de vacío por los transistores y se consiguió así una reducción considerable del tamaño de los ordenadores, una reducción de la temperatura exterior y una reducción de costes, ya que los transistores están basados en un elemento muy abundante: el silicio.

Aunque el sistema de cableado para programar el ordenador se seguía manteniendo -sobre todo, inicialmente-, se empezó a utilizar el sistema de tarjetas perforadas para la realización de programas. Hacia el final de esta generación, aparecieron los primeros lenguajes de alto nivel (por ejemplo, el lenguaje FORTRAN[19]).

(19)FORTRAN: ***for**mula **tran**slator/ translation*.

Había mucha diferencia entre la velocidad de la CPU y la de los periféricos (tarjetas perforadas), de manera que la CPU estaba mucho rato sin trabajar.

2.4.3. Tercera generación (1964-1974)

A partir del año 1964, con la llegada del ordenador IBM 360 se inició una nueva generación de arquitecturas de ordenadores de la mano de los **circuitos integrados**, capaces de integrar en espacios muy pequeños un gran número de transistores. A lo largo de los años, esta integración ha pasado de centenares a los millones de transistores que hay hoy día.

A partir de esta generación, y más concretamente a partir de la aparición del IBM 360, los programas los controla un sistema operativo. Se mejora la velocidad y la efectividad de los periféricos.

Para solucionar el problema de inactividad de la CPU cuando se realizan operaciones de periféricos de entrada y salida, los sistemas operativos de estas generaciones aceptaban realizar **programación concurrente**, que consiste en utilizar la CPU para hacer operaciones de ejecución mientras los periféricos están trabajando. De esta manera, se aprovecha más el tiempo de los diferentes componentes de la máquina. Hasta ahora, mientras se tenía que hacer una operación de los periféricos de entrada/salida, la CPU se mantenía detenida.

2.4.4. Cuarta generación (1974-actualidad)

El componente que ha caracterizado a esta generación ha sido el **microprocesador** o **CPU**. La ventaja de este componente ha sido el aumento de velocidad en la ejecución de las órdenes de los programas, el ahorro de consumo, la reducción espectacular del tamaño, la reducción de coste y el aumento en la potencia de cálculo.

Con la creación de los sistemas integrados (LSI/VLSI), los chips contienen miles de transistores en un espacio muy reducido.

Tipo de circuitos integrados	Número máximo de transistores	Equivalencia en número de puertas lógicas
LSI[20]	1.000 – 10.000	100-1.000
VLSI[21]	10.000 – 100.000	1.000 – 10.000

Ventas de ordenadores

A partir del año 1981, las ventas de ordenadores empezaron a crecer exponencialmente de un año al otro.

(20)LSI: *large scale integration*.

(21)VLSI: *very large scale integration*.

El número máximo de transistores en estos circuitos integrados puede parecer muy elevado, pero tenemos que pensar que esto ha evolucionado mucho y actualmente ya hablamos de millones de transistores.

Al principio de esta generación, en los ordenadores predominaban dos sistemas operativos: **MS-DOS**, creado por Microsoft para el ordenador IBM-PC con CPU Intel 8088, y **Unix**, para ordenadores que utilizaban la CPU Motorola 68000.

Con la aparición de ordenadores interconectados por redes y por Internet, aparecieron los sistemas operativos en red y los sistemas operativos distribuidos.

Hoy día, los sistemas operativos más utilizados son los siguientes:

- Sistemas operativos en red o los distribuidos.
- Sistemas operativos multiprogramados/multiusuario.
- Sistemas operativos en tiempo real: procesan en poco tiempo informaciones exteriores procedentes normalmente de sensores. Por ejemplo, utilizados en simuladores, control de vuelo en aviones, etc.

Tabla resumen de los tipos de tecnología utilizada en cada una de las diferentes generaciones de sistemas operativos

Generación	Tipo de tecnología
Primera	Válvulas de vacío
Segunda	Transistores
Tercera	Circuitos integrados
Cuarta	LSI/VLSI

2.4.5. Ejemplos de sistemas operativos más significativos en las últimas décadas

Para acabar, podemos ver algunos ejemplos de los sistemas más utilizados hoy día por distintas características: Unix, Windows, Linux, MAC, Novell y una pequeña introducción a los sistemas distribuidos.

Windows

En sus versiones XP y Vista, éste es el sistema más conocido hoy día. Pertenece a la empresa Microsoft del multimillonario Bill Gates.

Windows es un sistema operativo de 32 bits o 64 bits que permite trabajar en modo protegido, con multiárea real, con capacidad multimedia, interfaz GUI[22] orientada a objetos, ayuda interactiva, con posibilidades de trabajar en red, gestión de módem y fax, incorporación de tecnologías como OLE y *plug and play*.

[22]GUI: *graphic user interficie* ('Interfaz gráfica de usuario').

MS-DOS

Es el sistema operativo que dio origen a Windows XP, Vista o Windows 7. Se ejecutaba en línea de comandos y había que escribir cada instrucción y parámetros de uno en uno para que se ejecutaran los programas.

QDOS

En noviembre de 1980, Bill Gates y Paul Allen compraron un sistema operativo denominado QDOS (*quick and dirty operating system*) en Tim Patterson. Compraron este sistema para venderlo a IBM con el objetivo de que la compañía, en agosto de 1981, pudiera presentar su nuevo ordenador personal, PC. El sistema aparecía con el nombre de PC-DOS, antecedente de Windows.

Desde DOS hasta Windows XP ha habido mutaciones del sistema más o menos significativas. Hoy día, es el sistema más extendido en los usuarios de PC.

Windows 3.1 i 3.11

Fue el primer sistema de la compañía Microsoft que ya tenía un sistema de ventanas parecido al de MAC muy sencillo que permitía hacer tareas de manera visual y amigable.

Se presenta Windows 3.0 en 1990 como en entorno operativo. Esto quiere decir que Windows se comunica con el usuario y con el MS-DOS. Por otra parte, MS-DOS se entiende con Windows y con el ordenador.

Con Windows 3.1, el entorno de ventanas ya utiliza fuentes TrueType (lo que se ve es lo que se imprime), multimedia, enlace e inserción de objetos (OLE[23]), y la capacidad de que una aplicación reinicie la máquina. Esta versión evolucionó hacia la versión 3.11.

[23]OLE: *object linking and embedding* ('enlace e incrustración de datos').

Windows NT

Windows NT aparece en 1993 y se produce el salto de los 16 a los 32 bits en la palabra de programación del sistema. Esto permitía manejar distintos programas de manera paralela y no tener que detener a uno para activar el otro, multiproceso real, seguridad y protección de memoria.

No tenía una orientación a usuarios privados, sino a empresas que tuvieran que realizar un trabajo en colaboración en red. Estaba diseñado para estaciones avanzadas de trabajo y para servidores, además de para tomar ventaja de los procesadores más avanzados de Intel y RISC. No funcionaba sobre MS-DOS, sino que ya era un sistema operativo en sí mismo.

Windows 95

Con Windows 95 MS-DOS arrancaba el sistema, pero a continuación, de manera manual o automática, aparecía la interfaz gráfica de Windows 95 (Chicago).

Ahora se podía manejar el ordenador a partir de seleccionar los ficheros con el puntero y hacer clic, arrastrarlos y soltarlos y obtener información y funciones asociadas con el botón derecho del ratón, todo un abanico de interactividad. Con iconos que permitían un acceso elemental a los programas.

Desaparición de la línea de comandos

Habían desaparecido los tiempos oscuros (nunca mejor dicho) de la línea de comandos, cuando la incertidumbre asaltaba al usuario frente a la pantalla vacía.

Windows 98

El 25 de junio de 1998 aparece Windows 98 que, aunque hace mucha publicidad, realmente tiene pocos cambios con respecto a la última versión del 95 OSR-2.

Las pocas novedades que presentaba esta versión consistían en la posibilidad de actualizar el sistema por medio de Internet de una manera transparente para el usuario; mejoras en el escritorio activo y soporte para USB, DVD y para el uso de distintos monitores en un único PC.

Juicio Microsoft

Éste es el sistema que dio lugar al famoso juicio de Microsoft, por el hecho de haber integrado el navegador de Internet y el sistema de manera que el usuario no avisado optaba por Explorer en vez de buscar opciones para la navegación de otras compañías.

Windows Millenium

En este caso, es la versión final de todo el proyecto W95. Se trata de un sistema orientado a los PC caseros con todas las prestaciones de un sistema operativo para redes procedente de la vertiente NT. Este sistema incorpora realmente pocas mejoras en cuanto a sistema operativo, y muchas en la vertiente del entorno amigable y la orientación a tareas que será definitiva en XP. Tiene mejoras *plug and play*, incorpora la gestión de placas inteligentes para hibernar y suspender y permite compartir cuentas de Internet.

El resto o ya estaba en las versiones posteriores del 98, o proviene de las mejoras de entorno en red de NT. Sobre todo, la novedad que puede presentar este sistema es más obra de diseñadores que de informáticos. Por otra parte, incorpora mejoras multimedia pensadas para el usuario no profesional.

Windows 2000

Es el heredero de NT, aunque no ha tenido una vida muy intensa. En realidad está pensado para el mundo empresarial y destaca en la implementación de redes, pero presenta muchas incompatibilidades con programas habituales en el uso particular o casero. Por este motivo no se llegó a utilizar mucho, ya que se explotó en las mismas fechas que Millenium, que presentaba grandes ventajas.

Existe tanto la versión de estación de trabajo (para una sola máquina u ordenador) como la versión de servidor (para distintos ordenadores).

Windows XP

Windows XP de nombre Whistler es el primer salto cualitativo desde que se pasó de Windows 3.X a Windows 95. Este sistema XP es el heredero de Windows 200 y Windows ME con núcleo y arquitectura de Windows NT. Fue lanzado el 25 de octubre del año 2001. Hasta este momento, en todas las versiones, Windows todavía estaba más o menos oculto del sistema operativo MS-DOS. En XP ya ha desaparecido MS-DOS, aunque es perfectamente accesible desde el escritorio del sistema. No obstante, al no tener las limitaciones de este antiguo sistema, se ha podido programar en 32 bits y 64 bits un sistema operativo orientado al mismo tiempo a los usuarios aficionados y a las grandes empresas.

Fue la fusión de un sistema operativo único basado en NT con la funcionalidad de MS-DOS.

Reduce el tiempo de arranque y el de reinicio, a costa de aumentar el de parar-lo. Por fin, tiene un sistema multiárea completo en el que cada programa tiene su propia parte de memoria. Finalmente, desaparecen las habituales pantallas azules que han hecho tan famoso este sistema operativo.

Windows XP

Une las dos líneas que han seguido los sistemas operativos hasta ahora: la seguridad y firmeza en los intercambios de Windows 2000 con la compatibilidad y la facilidad de uso de la línea Windows 98/ME.

Presentó como novedades un nuevo ambiente gráfico, la instalación de aplicaciones y controladores sin necesidad de reinicio, el uso de diferentes usuarios al mismo tiempo, *cleartype* para leer mejor los textos, el escritorio remoto y soporte para módems ADSL y conexiones *wireless*.

Con la definitiva implantación de Internet en la interfaz y el núcleo del ordenador, es un sistema pensado para que se actualice continuamente por medio de Internet en un concepto nuevo con respecto a la vida y el mantenimiento del sistema: meses después de haber cargado el sistema operativo, éste no será el mismo, vistas las múltiples actualizaciones que se pueden realizar.

La interfaz amigable del sistema está ahora plenamente orientada a tareas en vez de estar orientada a la gestión de archivos. Esto significa que el sistema operativo es el más transparente para el usuario, encenderá el sistema y por medio de la interfaz, en pocos clics de ratón, ordenará lo que quiere hacer y con qué elementos, y no se tendrá que preocupar de nada más.

Ergonomía XP

Puede parecer muy natural trabajar con ventanas a quien se acaba de comprar el ordenador, pero si lo hemos tenido cerca con una pantalla oscura y una línea de comandos en el antiguo MS-DOS, la interfaz de XP parece como mínimo mágica, aparte de sencillamente ergonómica.

Se presentó en dos ediciones: HOME para particulares y Professional para empresas y profesionales.

El sistema de actualización se ha vehiculado con lo que se llama Service Pack; en él, se incluyen todas las actualizaciones hasta la fecha, además de algunas nuevas aplicaciones o versiones nuevas de aplicaciones incluidas ya en el núcleo primero.

El Service Pack 1 incluye la posibilidad de soporte para USB 2.0 y LBA de 48 bits, para discos duros de gran capacidad. El Service Pack 1, corrección del anterior, quita la Máquina Virtual Java de Microsoft por un problema con Sun Microsystems. El Service Pack 2 aporta correcciones al Service Pack 1 con la idea de dar mayor seguridad en el sistema operativo. El Service Pack 3 contiene actualizaciones independientes de Windows XO y características tomadas de Windows Vista.

El soporte de la compañía para el núcleo del sistema finalizó en el 2004, aunque el soporte para los Service Pack puede alargarse hasta el 2014.

Este sistema ha recibido muchas críticas por la integración en el sistema de aplicaciones que hacen competencia a terceros que quedan fuera del mercado, pues el usuario no se atreve a de duplicar aplicaciones, también por su predisposición a infectarse con virus. Por ello, hay que estar continuamente incorporando actualizaciones.

Windows Vista

Hecho por ordenadores de sobremesa, portátiles, *tablets* y centros multimedia, fue lanzado en el 2007. Decepcionó por los requerimientos y prestaciones que pide del ordenador y los problemas de compatibilidad con software y controladores.

Las novedades que aporta ni son notables ni son originales: Aero, la nueva interfaz gráfica con semitransparencias y efectos visuales en las ventanas; Internet Explorer 7 (desde ahora sustituido por Internet Explorer 8 con navegación por pestañas, *antiphishing* y modo protegido); Windows SIdebar, con pequeños programas o *gadgets* con diferentes herramientas, menudas y funcionales; Windows Media Player 11; compatibilidad con Extensible Firmaware Interface (EFI), con GPT en lugar de MBR; ventanas dibujadas de forma vectorial; una nueva API, WinFX; capacidad nativa para grabar DVD; Windows PowerShell interfaz de línea de mandos; RSS integrado; mejoras de restauración de sistema, Windows Comunication Foundation, sistema de comunicaciones unificado; Windows Defender, *antispyware*; Windows Mail en lugar de Express; tecnología de caché de disco Windows ReadyBoost; protección de datos con BitLocker Drive Encryption; User Account Control para tareas administrativas con una ventanilla de confirmación tipo Linux; Sunc Centre, para sincronizaciones; Windows Sotfware Protection Platform sustituye a Windows Genuine Advantage.

Por último, hay que señalar que carga un 15% más rápido, entra en suspensión en 2 segundos y reduce a un 50% la necesidad del reinicio del sistema.

Hay distintas versiones: Starter, HOma-Basic, HOmePremium, Business, Enterprise y Ultimate. Van todas con el mismo DVD de carga, pero según el serial se carga una u otra. Las versiones básicas pueden ser actualizadas en las superiores mediante Windows Anytime Upgrade.

También dispone de sus Service Pack por actualización del sistema, de momento el 1 y el 2.

Ha recibido evaluaciones muy negativas debido a su bajo rendimiento, la aparición continuada de la ventana de confirmación de acciones y la incompatibilidad con la mayor parte de accesorios empresariales, de forma que sirve para usuarios domésticos, pero da muchos problemas para la adecuación en el software y el hardware anteriores, los cuales funcionaban perfectamente con XP.

Windows 7

Nombres clave: Blackcomb y Vienna. Se trata de un sistema, todavía no acabado, que se presenta como una actualización del núcleo NT 6.0. Las metas que persigue son mejorar la interfaz, alcanzar un sistema más ligero, estable y rápido, incorporar capacidades táctiles e implementar un sistema de redes domésticas. Tendrá versiones de 32 y 64 bits y tratará de ser más simple y evitar todos los errores cometidos en Windows Vista, que son muchos.

Asimismo, intentará mejorar el rendimiento del sistema trabajando sobre: la ocupación de memoria, la utilización de la CPU, operaciones de entrada y salida a disco, arranque, cierre y reposo, el rendimiento del sistema base y la ocupación de disco por parte del sistema.

Windows .NET Server

Este sistema operativo es una evolución del Windows 2000 Server y del Windows XP, con intención de dar respuesta a la demanda de los usuarios con respecto a páginas web y a comunicaciones microinformáticas.

OS-2

Este sistema fue un proyecto conjunto de Microsoft e IBM que, por razones complejas, acabó a manos de IBM mientras Microsoft continuaba desarrollando su MS-DOS/Windows.

Sus características eran compatibles con DOS en ordenadores 800020x86 (nombres de CPU de la casa Intel), multitarea, memoria virtual y servicios de red de área local.

Tenía dos modos de trabajo: el sincrónico y el asíncrono. Permitía trabajar con particiones en un solo disco y mantener sistemas de archivos en cada partición. Disponía de una interfaz gráfica.

Compatibilidad con DOS

Con el objetivo de mantener la compatibilidad con DOS, la versión 1.0 de OS/2 era muy parecida a la de este sistema operativo. Más adelante, en las versiones 2.x, mejoró el sistema de archivos.

En OS/2, los procesos se pueden suspender para darle su turno de ejecución a otro diferente y pueden estar divididos en cadenas (*threads*). También es posible que un proceso genere un proceso hijo. Igualmente, puede crear *pipes all mode* de Unix. Disponía de carga dinámica de librerías: se cargan cuando el programa no se ejecuta dentro del mismo programa. Soportaba segmentación y paginación en el manejo de memoria. Con respecto al modelo de E/S, este sistema intentaba la compatibilidad con los programas de DOS por medio de un supervisor de dispositivos, lo cual implicó algunos problemas de diseño.

Unix

El sistema evoluciona en los Laboratorios Bell de AT & T, y más adelante se expandió por las universidades, hasta llegar a la creación de las versiones más importantes, las de la Universidad de Berkeley y el sistema V.

Después, el estándar IEEE[24] utilizó un algoritmo consistente en revisar las llamadas al sistema de las dos versiones (System V y BSD), y las que eran iguales las definió como estándares, de modo que surgió la definición *portable operating system for Unix*, o Postix.

(24)IEEE: *Institute of electrical and electronics engineers* ('Instituto de Ingenieros Eléctricos y Electrónicos').

IBM, DEC y Hewlett-Packard lanzaron su propia versión de Unix denominada OSF/1 (Open Software Foundation). Esta versión cumplía los estándares del IEEE y tenía, además de un sistema de ventanas (el X11), una interfaz mejorada para los usuarios (MOTIF) y definiciones para cómputo distribuido (DCE) y administración distribuida (DME).

Por su parte, SunOS de Sun Microsystems ofrecía otra versión con una interfaz amigable y un conjunto de librerías para crear aplicaciones con interfaz gráfica denominada SunWindows o SunVIEW. AT&T formó, junto con Sun Microsystems y otras compañías, Unix International y su versión de Unix, y así provocó que ahora se manejen estas dos corrientes principales en Unix.

El sistema de archivos de Unix tiene una organización jerárquica que parte de una raíz "/". También ofrece un poderoso conjunto de comandos y llamadas al sistema. La protección de archivos en Unix se ha definido como una cadena de permisos de nueve caracteres (RWX-RWX-RWX), basada en la idea de propietario del archivo/grupo de propietarios / resto de los usuarios.

El núcleo del sistema se denomina *kernel*, y se encarga de las interrupciones, los manejadores de dispositivos de bajo nivel, del manejo de la memoria, las llamadas al sistema, la planificación de procesos, el entubamiento, la paginación y el intercambio y el manejo de discos y del sistema de archivos.

El sistema permite crear *pipes* entre procesos, contabilizar el uso de CPU por proceso y una pila común para todos si se necesitan ejecutar en modo privilegiado. Utiliza el manejo de memoria virtual con paginación por demanda y combinación de segmentos paginados, con páginas de tamaño fijo.

Los dispositivos en este sistema se tratan como archivos a los cuales se accede por medio de descriptores de archivos, cuyos nombres se encuentran generalmente en el directorio '/dev'.

Linux

Linux es un sistema creado a partir de Unix, exactamente de Posix, adaptado a los PC de tabla. Éste es hoy día prácticamente el único sistema operativo que puede competir con Windows en el entorno PC.

Se trata de un sistema de libre distribución creado por la comunidad de Internet, a partir de un núcleo que creó **Linus Torvalds**. En un principio, se pusieron las fuentes del núcleo a disposición de la comunidad de Internet y la comunidad fue creando y añadiendo poco a poco programas, aplicaciones, gestores, controladores, librerías y conjuntos de librerías.

Lo que los aficionados al sistema implementaron se ha ido añadiendo al núcleo primero, y el sistema ha ido creciendo y teniendo cada vez más capacidad e inteligencia en el sentido de que no necesita a ningún usuario experimentado para su gestión.

Algunas compañías o equipos han decidido trabajar por su cuenta y crear sobre el núcleo de Linux una distribución. Esta distribución ya no incluye módulos separados, sino que es un paquete que instala Linux en nuestro ordenador, tal y como pasa con Windows. Las distribuciones van desde las más básicas pero potentes como Devien, hasta las más complejas y fáciles de instalar como RedHat, SuSe, Mandrake o Caldera que prácticamente se instalan sin casi hacer configuraciones ni adaptaciones.

Aunque está pensado para PC de mesa, es capaz de correr sobre DEC Alphas, SUN Sparcs, máquina M68000 (como Atari y Amiga), MIPS y PowerPC.

Tiene todas las características importantes de Unix, cumple los estándares IEEEPosix 1 System V y BSD, tiene multiusuario y multiárea real, acceso a memoria virtual, librerías compartidas, enlace dinámico, carga por demanda, ejecutables compartidos, gestión de memoria y protocolos TCP/IP.

Soporta distintos sistemas de ficheros, entre éstos, MS-DOS, Windows, OS2 e ISO9660 (CD), de manera que, montando unidades con otros sistemas de archivos procedentes de otros ordenadores, es posible acceder a los diferentes discos y tener compatibilidad en determinados archivos. Actualmente, dispone de controladores para casi cualquier elemento hardware de uso corriente, aunque es más difícil encontrar productos menos comunes.

En principio, el sistema se maneja en línea de comandos, la pantalla oscura y enseguida intermitente que nos pide una entrada de texto. No obstante, con enorme facilidad, si lo preferimos, podemos acceder a un sistema amigable de ventanas al modo de Windows o MAC, e incluso elegir entre distintos sistemas de ventanas con diferentes prestaciones y diseños según nuestras preferencias, como pueden ser GNOME[25] o KDE[26].

[25]GNOME: ***G****NU* ***n****etwork* ***o****bject* ***m****odel* ***e****nvironment*.

[26]KDE: ***Kd****esktop* ***e****nvironment*.

Se distribuye bajo la licencia GNU, que trata de garantizar la libertad para compartir y retocar el software libre, y garantizar que el software es libre para todos sus usuarios.

Ventajas

Esto presenta una ventaja muy grande sobre el sistema Windows. Cuando se adquiere el software Windows, no se adquieren ni las fuentes donde esté escrito el programa ni los ejecutables que se tienen en el ordenador, sino que más bien se tienen en usufructo y para una única máquina.

La idea no es que el software tenga que ser gratuito y de uso libre para todo el mundo, sino que una vez adquirido por el usuario no tiene restricciones: por una parte, se dispone del código fuente (archivos de texto con los que se han generado los binarios ejecutables) y por la otra, de la libertad para modificarlo según los intereses concretos de los que han adquirido los programas. Esta licencia se aplica a los programas surgidos a la sombra de la Fundación de Software Libre.

Además, al no disponer de las fuentes para hacer cualquier adaptación que nos interese o alguna modificación necesaria para hacer mejor nuestro trabajo, cualquier problema tiene que pasar por el equipo de informático de Windows, lo cual representa un coste y una dependencia no deseables.

Por otra parte, cuando se compra un sistema basado en Linux, también se obtienen las fuentes y la documentación para hacer la gestión del sistema. De hecho, parece que el negocio de Linux ha sido precisamente las distribuciones y la documentación; evidentemente, no es un negocio tan lucrativo como el de Windows, pero demuestra que no es necesario tanto oscurantismo con el software como propone la compañía de Bill Gates.

La competencia

Hoy día, Linux está haciendo la competencia seria a Microsoft, no como a Linux, sino por medio de sus diferentes distribuciones, como RedHat, SuSe, Mandrake o Caldera.

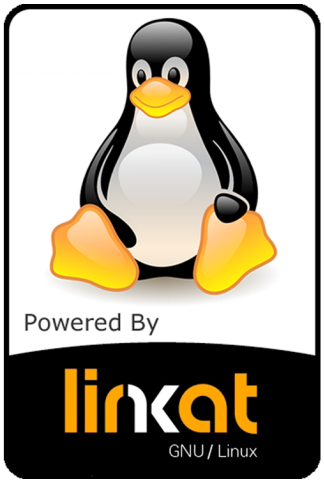
La comunidad e Internet han creado, conjuntamente con algunas compañías, bastantes programas, muchos totalmente gratuitos, que hacen de Linux una alternativa viable hoy día con respecto al otro sistema MSWindows. Por una parte, las distribuciones son de muy fácil instalación, ya que en otro momento esto hizo que Linux se reservara a los entendidos en informática y, por otra, hay muchos programas y aplicaciones que son hasta un punto determinado equivalentes a los profesionales y de pago que se distribuyen para el sistema Windows.

Recordemos que el sistema Windows se tiene que pagar y se debe tener una copia autorizada para que funcione en nuestro ordenador, y lo contrario es un delito, aunque sea habitual. En el caso de Linux, con las facilidades que hay hoy día, no nos tendremos que sentir delincuentes por escribir en la máquina.

En los últimos años, las diferentes autonomías han decidido implementar Linux tanto en su administración como en los centros educativos no universitarios. Aquí podéis encontrar una recopilación de las diferentes distribuciones que han surgido en los últimos años.

- Linkat: distribución GNU/Linux del departamento de enseñanza de la Generalitat Catalana. Está basada en la distribución SUSE Linux Enterprise, es decir, en paquetes RPM, y utiliza los escritorios GNOME, KDE y XFCE.

Incluye muchas aplicaciones de comunicación, creación y software educativo.



- MAX: distribución Debian GNU Linux basada en Ubuntu de la Consejería de Educación de la Comunidad de Madrid. Utiliza escritorios GNOME, KDE y/o XFCE. Está especialmente volcada en los usuarios con problemas de accesibilidad. Dispone de un gran conjunto de herramientas elaboradas a partir de software libre para la ONCE.

- Molinux: distribución GNU/Linux oficial de la Junta de Comunidades de Castilla-La Mancha, basada en Ubuntu. De momento, es un sistema operativo general, pero en el futuro habrá versiones modulares adaptadas a usos específicos. Como peculiaridad, tenemos que decir que las versiones reciben el nombre de personajes del *Quijote*, copiando el rasgo Debian, que denominaba a sus versiones con personajes de *Toy Story*.

- Lliurex: distribución oficial de la Consejería de Educación de la Generalitat Valenciana que utiliza únicamente el escritorio GNOME. Su objetivo principal es la introducción de las nuevas tecnologías en el sistema educativo. Está basada actualmente en Ubuntu, aunque empezó haciéndolo en Debian. Está especialmente diseñada para ser instalada en las aulas de los centros educativos.

- GnuLinex: distribución de la Consejería de Economía, Comercio e Innovación de la Comunidad Autónoma de Extremadura y basada en Debian

GNU/linux con escritorio GNOME. Es la primera que se hizo y el modelo en el que se han basado otras autonomías. Ha recibido gran cantidad de premios.

- Guadalinex: distribución promovida por la Junta de Andalucía. Inspirada en GnuLinex, y ahora en Ubuntu, aunque antes directamente en Debian. Hay muchas variaciones según el usuario al que va dirigida: bibliotecas, centros de día de personas mayores, centros educativos, etc.

Novell Netware

Novell Netware es una red de PC con un servidor dedicado. Nació en 1983 como un sistema operativo para una red propietaria de Novell (S-Net), con topología de estrella y un servidor de archivos basado en el microprocesador MC68000 de Motorola.

Se pretendió crear un sistema operativo que funcionara de manera más efectiva en modo multiusuario. A causa de esto, Netware se escribió específicamente para el hardware de los sistemas basados en la familia 8086, en lugar de estarlo para DOS y su sistema de E/S. Tenía un *shell* en torno a DOS, interfaz que permitía que los usuarios de las estaciones de trabajo trabajaran con DOS.

En un primer momento, hubo cuatro versiones básicas de Netware 286:

- **ELS**[27] **Netware Level I.**

(27)ELS: *entry level system.*

- **ELS[27] Netware Level II.**

- **Advanced Netware.**

- **SFT[28].**

(28)SFT: *system fault tolerant.*

En general, Netware 286 utiliza un sistema de directorios de archivos parecido al sistema jerárquico del DOS 2.0, y proporciona distintos sistemas de protección ante algún fallo que se presente en el servidor de archivos o en el disco duro. La seguridad de Netware se basa en perfiles de usuario: se utiliza el sistema de claves de acceso. Los directorios de Netware tienen ocho tipos diferentes de derechos, varios de los cuales no soportan el DOS. Estos derechos son: *read, write, open, create, delete, parental, search* y *modify*.

Por otra parte, también se hizo Netware 386, una versión completamente nueva escrita específicamente para los microprocesadores 386 y 486.

Novell, hoy día, es una arquitectura denominada *sistemas abiertos Netware*. Esta arquitectura tiene los objetivos siguientes: disponer de los servicios ofrecidos por Netware en plataformas ampliables, que sea independiente del protocolo soportando los estándares importantes.

MAC OS X

Un sistema operativo muy extendido también entre el parque de PC es Macintosh. Este sistema lo creó la empresa Apple y, curiosamente, aunque no está tan extendido como los sistemas Windows, fue el primero que basó la manera de trabajar en un entorno de ventanas que después se ha ido plasmando también en los sistemas de Microsoft.

Está basado en la Berkeley Software Distribution (BSD) de Unix de los años setenta. De hecho, las librerías y utilidades de MAC USTED proceden de FreeBSD. Los servicios de red están basados en el estándar BSD TCP/IP.

La historia de este sistema es inseparable del ordenador y la compañía Apple, ya que está pensado para correr sobre este tipo de ordenador.

El Apple II fue el primer ordenador personal equivalente funcionalmente a las normas actuales. Se lanzó en 1977, equipado con teclado, programación externa y monitor de color. Fueron creados en una cochera por Steve Wozniak y Jobs.

Lanzamiento de Apple

El Apple se lanzó en 1976. Paradójicamente, se considera como el producto que provocó la revolución de los ordenadores personales.

El sistema disponía de un menú en la parte superior con opciones para las aplicaciones, ventanas donde se mostraba el entorno de trabajo, una papelera para guardar los ficheros antes de su destrucción y, además, iconos, el elemento de interacción más directa; para acceder a todos estos elementos con facilidad, se introdujo un nuevo periférico, el ratón.

Inspiración Xerox

Este ordenador fue el primero que incorporaba una interfaz gráfica. La idea de interfaz gráfica fue copiada de los laboratorios Xerox.

Es un sistema basado en Unix, tiene incorporadas herramientas de desarrollo que permiten crear programas para el sistema y dispone de una arquitectura modular basada en cuatro capas: el *core* de Unix, el sistema gráfico, el entorno de trabajo para aplicaciones y la interfaz de usuario.

Sistemas distribuidos

Por otra parte, hoy día los sistemas más avanzados son los que denominamos *sistemas distribuidos*, los cuales mencionaremos a continuación.

En torno a la década de los ochenta, a la sombra del desarrollo de microprocesadores poderosos y económicos, junto con la implementación progresiva de redes de área local (LAN) de alta velocidad aparecen los sistemas distribuidos. En estos sistemas, en contraste con los sistemas centralizados, los usuarios pueden acceder a una gran variedad de recursos, y tienen distintas CPU conectadas entre sí que trabajan de manera conjunta.

Esto presenta determinadas ventajas:

a) Economía, pues la proporción precio/rendimiento es mucho mejor. Están diseñados para que muchos usuarios trabajen conjuntamente.

b) Fiabilidad; si una máquina se descompone, sobrevive el sistema.

c) Escalabilidad, se puede añadir poder de cómputo en pequeños incrementos.

d) Comunicación, compartir determinados datos, recursos, programas y periféricos y mejor comunicación entre las personas y más flexibilidad con grupos de trabajo.

Hay muchos tipos de sistemas distribuidos, ahora bien, básicamente hay dos modelos:

Basados en multiprocesadores	El conjunto de procesadores reciben demandas de los usuarios.	Tienen memoria compartida.
Basados en multiordenadores	Cada usuario dispone de su equipo, de modo que la mayor parte del trabajo se hace de manera local.	No tienen memoria compartida.

Si las máquinas y los usuarios son independientes entre sí, hablamos de sistemas operativos débilmente adaptados. Cada usuario tiene una estación de trabajo para su uso exclusivo y su sistema operativo, y los requerimientos se resuelven localmente, en la máquina del usuario. Sólo se puede utilizar una máquina aunque se tenga un sistema de archivos global compartido a modo de servidor de archivos, accesible desde todas las estaciones de trabajo.

En este entorno, el sistema distribuido se encarga de controlar las estaciones de trabajo en el individual, la comunicación entre los servidores y los servidores de archivo. No es necesario que todas las máquinas tengan el mismo sistema operativo. Los sistemas que siguen este esquema se denominan *sistemas operativos de red*.

NTFS[29]: es uno de los más conocidos. Surgió para Unís, pero se amplió en otros sistemas operativos. Se caracteriza por permitir que un grupo de clientes y servidores compartan un sistema de archivos comunes. Además, permite que cada máquina sea un cliente y un servidor al mismo tiempo, y exporta uno o varios de sus directorios (y subdirectorios dependientes) para el acceso por parte de clientes remotos, de modo que crea un sistema heterogéneo en el que los clientes y servidores podrían ejecutar distintos sistemas operativos en hardware diferente.

(29)NTFS: *network file system*.

Por otra parte, los multiordenadores son un ejemplo de software fuertemente adaptado en hardware débilmente adaptado, de manera que se crea la ilusión de que toda la red de ordenadores es un solo sistema de tiempo compartido que se ejecuta en una colección de máquinas sin memoria compartida, pero que aparece delante de sus usuarios como un solo ordenador. Esto permite una comunicación global entre los procesos siguiendo un esquema global de protección, aunque la administración de procesos tiene que ser la misma. Tienen la misma interfaz de llamadas al sistema y sistema global de archivo.

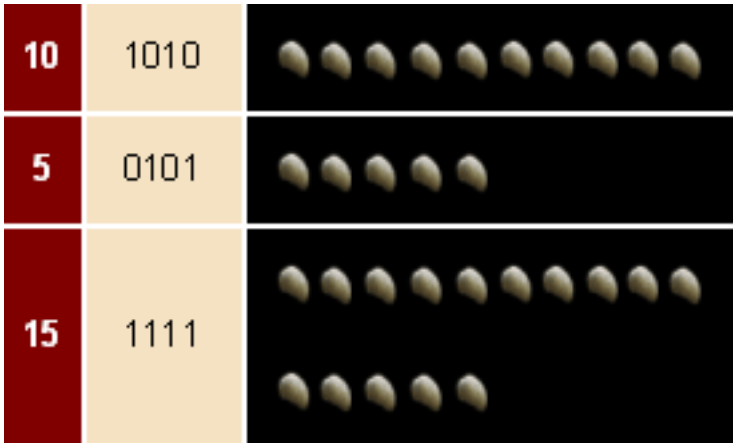
Finalmente, podremos encontrar casos de software fuertemente adaptado en hardware, muy adaptado a los ejemplos más comunes de propósito general, que son los multiprocesadores: como un sistema de tiempo compartido, pero con distintas CPU en vez de una sola. Mantienen la imagen de un sistema único y tienen una sola cola para ejecución contenida en la memoria compartida. Todos los programas se almacenan en la memoria global compartida, que tendrá un sistema de archivos tradicional.

Los aspectos clave en el diseño de sistemas operativos distribuidos son los siguientes: la transparencia, es decir, percibir que la colección de máquinas conectadas son un sistema de tiempo compartido de un solo procesador, de manera que no sea visible la existencia de distintos procesadores; la flexibilidad, que implica alta modularidad y una interfaz bien definida con cada servicio que es igual de accesible para todos los clientes, y además debe ser fácil implantar, instalar y depurar nuevos servicios; fiabilidad, en el sentido de que si una máquina se equivoca, alguna otra se debe encargar del trabajo; rendimiento, ya que no debe parecer peor que su ejecución en un único procesador; y escalabilidad para poder utilizar centenares de miles e, incluso, decenas de millones de usuarios conectados.

3. Apéndice I

Transcribir signos de notación binaria a decimal no tiene mucha dificultad si seguimos el criterio siguiente.

Imaginamos 15 en decimal, que no es otra cosa que la suma de 10 + 5; en términos binarios, 1010 + 0101:



En decimal, nosotros asignamos el valor notación *1 a las unidades, notación *10 a la segunda posición desde la derecha porque éste es el valor de esta posición (décimas) en este sistema, el de la tercera (centésimas) será notación *100 y así sucesivamente (1.000 + 500 + 30 + 5 = 1 * 1.000 + 5 * 100 + 3 * 10 + 5 * 1). Sin embargo, ¿cuáles son estos valores en la representación del sistema binario?

Para realizar o entender la representación en sistema binario, debemos comprender lo siguiente:

La posición primera a la derecha puede tener notación 0 ó 1 y valor también 0 ó 1; la segunda posición, que le sigue a la izquierda, también puede tener la notación 0 ó 1, pero ahora su valor será 0 ó 2; la tercera posición (equivaliente a las centenas) continúa teniendo sólo las notaciones 0 ó 1 pero ahora sus valores son, en cada caso, 0 ó 4.

Es fácil entender el proceso: en decimal, cada posición corrida hacia la izquierda multiplica su valor por la base, es decir, 10. En binario pasa exactamente lo mismo pero puesto que la base es 2, la multiplicación se realiza por esta cantidad.

De este modo, nos podemos hacer una idea del valor de una posición y después sumar los valores de las diferentes posiciones que encontramos, de la manera siguiente:

P3	**P2**	**P1**	**P0**		**Posiciones = 4 (Px)**
2 * 2 * 2	2 * 2	2	1		**Base = 2**
$2^3 = 8$	$2^2 = 4$	$2^1 = 2$	$2^0 = 1$	Total	**Valor = 2^n**
1 * 8 +	**1** * 4 +	**1** * 2 +	**1** * 1 +	= 15	**Notación = 1111**
8+	4+	2+	1	=15	

Exactamente al igual que haríamos con el sistema decimal:

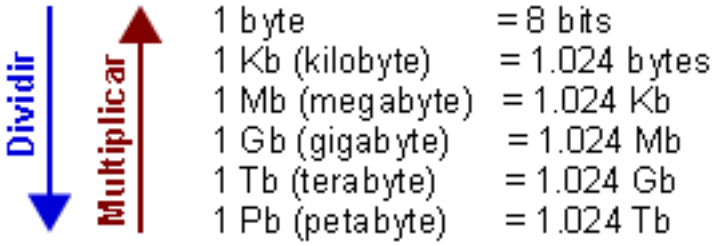
P3	**P2**	**P1**	**P0**		**Posiciones = 4(Px)**
10 * 10 * 10	10 * 10	10	1		**Base = 10**
10^3=1000	10^2=100	10^1=10	10^0=1	Total	**Valor = 10^n**
0 * 1000	**0** * 100	**1** * 10	**5** * 1	= 15	**Notación = 0015**
0 +	0 +	10 +	5	= 15	

Con facilidad, de estas dos tablas podemos extrapolar la manera en la que operaremos con cualquier sistema en base b con posiciones n, y de acuerdo con la notación x que figura en cada celda del número que tenemos que definir y transcribir.

n - 1	**n - 2**	**n - 2**	**n - n**	**Posiciones = n**
$b^{n-1} = b * b * b$	$b^{n-2} = b * b$	$b^{n-3} = b$	$b^{n-n} = 1$	**Base = b**
x* b*b*b	**x** * b * b	**x** * b	**x** * 1	**Notación = xxxx**

4. Apéndice II

A veces, estas informaciones vendrán con unidades demasiado pequeñas y las tendremos que pasar a unidades mayores o, al revés, serán unidades mayores que deberemos pasar a más pequeñas. Para hacer estas conversiones, nos fijaremos en la figura siguiente:



Nuestro problema es pasar una cantidad determinada de información en una unidad en concreto a otra unidad. La operación matemática que tendremos que utilizar para hacer la conversión vendrá dada por la **unidad que tenemos** y la **unidad que queremos**.

Ordenar unidades

Para que funcione este sistema de conversión de unidades, es muy importante que las unidades estén ordenadas de pequeña a grande. De hecho, es el mismo sistema que se utiliza para hacer cualquier conversión: longitud, volumen, etc.

Pasar 3 Mb a kb

Por ejemplo, queremos pasar **3 Mb a kb**. Matemáticamente hablando, se puede expresar con la equivalencia siguiente:

3 Mb = **?** kb

Tenemos la unidad Megabytes y queremos la unidad kilobytes. En este caso, tendremos que utilizar la operación de multiplicar, ya que la unidad Mb se encuentra, en la figura anterior, por debajo de la unidad kb.

Ahora ya sabemos que la operación que se debe utilizar será la multiplicación. Sin embargo, ¿por qué cantidad? Para responder esta pregunta, tenemos que mirar en las equivalencias de la tabla qué equivalencia habla de las unidades que estamos tratando, es decir, Mb y kb.

Sólo hay una equivalencia: 1 Mb = 1.024 kb. Por lo tanto, tendremos que multiplicar por 1.024.

Finalmente, haremos la operación matemática. En nuestro ejemplo, multiplicaremos lo que tenemos (3 Mb) por 1.024 (valor encontrado):

3 Mb = **3 x 1.024** kb = **3.072** kb

Si hacemos un repaso de los pasos que seguiría nuestra mente para hacer esta conversión, éstos serían:

1) **Pregunta**: 3 Mb = ? kb

2) **Operación que hay que realizar**: multiplicar, dado que Mb está por debajo de kb.

3) **Cantidad**: la equivalencia de la figura anterior que relaciona las unidades Mb y kb es la siguiente: 1 Mb = 1.024 kb, por lo tanto, la cantidad será 1.024.

4) **Operar**:

$$3\ Mb * \frac{1.024\ Kb}{1\ Mb} = 3 * 1.024\ Kb = 3.072\ Kb$$

Puede parecer un sistema bastante complicado, pero si pensamos que estamos trabajando con otras unidades, posiblemente nos parecerá más sencillo.

Conversión en euros

Por ejemplo, las unidades serán los euros. Si queremos pasar tres billetes de 5 euros a monedas de 1 euro, queda claro que tendremos que multiplicar 3 por 5. La operación de multiplicar se debe hacer porque en la tabla de unidades tendríamos la equivalencia de un billete de 5 euros por debajo de la moneda de 1 euro (es mayor) y la cantidad que hay que multiplicar nos la da la equivalencia: un billete de 5 euros = cinco monedas de 1 euro.

$$1\ billete\ 5€ * \frac{5\ monedas\ 1€}{1\ billete\ 5€} = 1 * 5\ monedas\ 1€ = 5\ monedas\ 1€$$

Pasar de bytes a kb

Hagamos un ejemplo que nos permita pasar de una unidad pequeña a una mayor: **1.048.576 bytes a kb**.

1.048.576 bytes = **¿?** kbytes

1) **Pregunta**: 1.048.576 bytes = ¿? kb.

2) **Operación que se debe realizar**: dividir bytes, ya que está por debajo de kb.

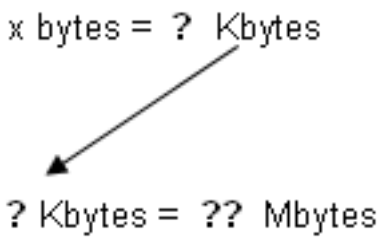
3) **Cantidad**. La equivalencia de la figura anterior que relaciona las unidades bytes y kb es la siguiente: 1.024 bytes = 1.024 kb, por lo tanto, la cantidad será 1.024.

4) **Operar**:

$$1.048.576\ bytes * \frac{1\ Kb}{1.024\ bytes} =$$

$$= \frac{1.048.576}{1.024} Kb = 1.024 Kb$$

Se pueden complicar los ejemplos si pedimos conversiones entre unidades más alejadas; por ejemplo, pasar bytes a Mbytes. El procedimiento será el mismo, pero más largo. Primero tendremos que pasar los bytes a kbytes y después, estos kbytes a Mbytes, es decir,



Ejercicios de autoevaluación

1. Buscad en Internet algunos ejemplos de aplicación de la domótica en las viviendas.

2. Llenad las celdas vacías de manera que podamos obtener las equivalencias en los diferentes sistemas numéricos:

Binario	Octal	Decimal	Hexadecimal
		33	
	56		
			0x4C
1011101			

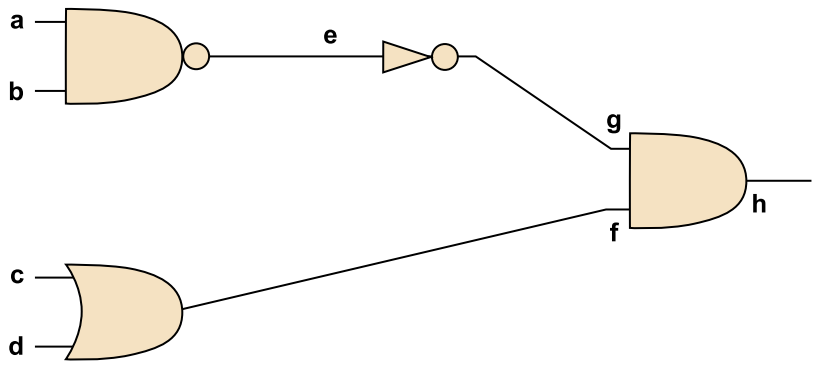
3. Indicad cuál es el tipo de periférico al cual corresponde cada uno de los periféricos siguientes:

- Pantalla
- Lector de CD-ROM
- Disco duro
- Escáner
- Unidad de disquetes

4. Indicad cuáles son las equivalencias:

20 GB = ? kbytes
10.240 Mb = ? Gb

5. Indicad cuál es la tabla de verdad del circuito digital siguiente:



6. Indicad si son verdaderas o falsas las afirmaciones siguientes:

El valor numérico después del 111 en sistema binario es el 1110.

a) Verdadero
b) Falso

Charles Babbage fue el creador del telégrafo óptico.

a) Verdadero
b) Falso

Las puertas OR dan como salida un 1 cuando las entradas valen 0.

a) Verdadero
b) Falso

7. Verificad con ejemplos prácticos que en sistema binario el hecho de poner un cero a la derecha equivale a multiplicar por dos. Por ejemplo, el valor 1011 es la mitad que 10110.

Binario	Decimal
1011	
10110	

8. ¿Quién escribió el artículo "Preliminary discussion of the logical design of an electronic computing instrument"?

9. ¿Cuáles son las partes del modelo de ordenador propuesto por J. Von Neumann?

10. ¿Cómo se llaman las partes que integran la unidad central de proceso?

11. ¿Cuáles son las ventajas principales de la memoria caché?

12. ¿Cómo se llama el registro que indica cuál es la próxima instrucción que la máquina tendrá que ejecutar?

13. Indicad si es verdadera o falsa la afirmación siguiente:

"Hoy día, disponer de muchos datos es disponer de información".

a) Verdadero
b) Falso

14. ¿Cuáles son los diferentes modos de señales de las unidades de almacenamiento?

15. ¿Cuáles son las partes de un sistema operativo?

16. ¿Cuántos kbytes de velocidad tiene un módem ADSL de 256 kbit que aprovecha toda su velocidad?

Solucionario

Ejercicios de autoevaluación

1.

- Sensores de humedad, humo y fuego que permiten activar la alarma o, incluso, llamar por teléfono al interesado.
- Sensores de presencia o intrusos.
- Para evitar a los ladrones, hay mecanismos que permiten subir y bajar las persianas, abrir o cerrar las luces de la casa, ver gráficamente determinadas partes de la casa, etc.
- Climatización automática.
- Mecanismos de asistencia o alerta médica mediante radiofrecuencia o telefonía.
- Interruptores que detectan movimiento. De esta manera, se consigue ahorrar electricidad cuando no estamos dentro de la habitación, la cocina, etc.
- Aprovechamiento de energías alternativas: eólica, solar, etc.
- Control remoto del teléfono, contestador, etc.

2.

Binario	Octal	Decimal	Hexadecimal
100001	**41**	33	**0x21**
101110	56	**46**	0x2E
1001100	**114**	**76**	0x4C
1011101	**135**	**93**	**0x5D**

3.

- Pantalla: salida
- Lector de CD-ROM: entrada, ya que este periférico sólo permite leer lo que ha grabado. Para grabar en un CD-ROM, necesitamos otro periférico (regrabadora).
- Disco duro: entrada y salida
- Escáner: entrada
- Unidad de disquetes: entrada y salida

4. 20 GB = 20.971.520 kbytes
10.240 Mb = 10 Gb

5.

a	b	c	d	e	f	g	h
0	**0**	**0**	**0**	1	0	0	0
0	**0**	**0**	**1**	1	1	0	0
0	**0**	**1**	**0**	1	1	0	0
0	**0**	**1**	**1**	1	1	0	0
0	**1**	**0**	**0**	1	0	0	0
0	**1**	**0**	**1**	1	1	0	0
0	**1**	**1**	**0**	1	1	0	0
0	**1**	**1**	**1**	1	1	0	0
1	**0**	**0**	**0**	1	0	0	0
1	**0**	**0**	**1**	1	1	0	0
1	**0**	**1**	**0**	1	1	0	0

a	b	c	d	e	f	g	h
1	**0**	**1**	**1**	1	1	0	0
1	**1**	**0**	**0**	0	0	1	0
1	**1**	**0**	**1**	0	1	1	1
1	**1**	**1**	**0**	0	1	1	1
1	**1**	**1**	**1**	0	1	1	1

6. El valor numérico después del 111 en sistema binario es el 1110.
b) Falso

Charles Babbage fue el creador del telégrafo óptico.
b) Falso

Las puertas OR dan como salida un 1 cuando las entradas valen 0.
b) Falso

7.

Binario	Decimal
1011	**11**
10110	**22**

8. J. Von Neumann en colaboración con Arthur W. Burks y Herman H. Goldstine.

9. Memoria principal, unidad central de proceso y unidades de entrada y salida.

10. Unidad aritmética y lógica, unidad de control.

11. Mucha velocidad y ahorro de accesos a la memoria principal.

12. Contador de programas (*program counter*).

13. b

14. Magnético, óptico y magnetoóptico.

15. Interfaz de usuario, sistema de ficheros, gestión de E/S, gestión de memoria, gestión de procesos y núcleo.

16. 256 kbits = 262.144 bits = 32.768 bytes = 32 kbytes.

Bibliografía

Bibliografía general

Anasagasti, Pedro de Miguel (2001). *Fundamentos de computadores I* (8.ª ed.). Madrid: Paraninfo.

Peterson, James L.; Silberschatz, Abraham (1994). *Sistemas operativos. Conceptos fundamentales*. Barcelona: Editorial Reverté.

Bibliografía de consulta

Anasagasti, Pedro de Miguel (1994). *Fundamentos de computadores*. Madrid: Paraninfo.

Boole, George (1972). *Investigación de las leyes del pensamiento sobre las que se fundamentan las teorías matemáticas de la lógica y las probabilidades*. Maracaibo: Universidad del Zulia.

Boole, George (1982). *Investigación sobre las leyes del pensamiento*. Madrid: Paraninfo.

Boole, George (1984). *El análisis matemático de la lógica*. Madrid: Cátedra.

Boyer, Carls B. (1999). *Historia de la matemática, Ciencia y Tecnología*. Madrid: Alianza Editorial.

Breton, Philippe (1989). *Historia y crítica de la informática*. Madrid: Cátedra.

Brooks, Rodney A. (2003). *Cuerpos y máquinas*. Barcelona: Ediciones B ("Sine qua non").

Ceballos Sierra, Francisco Javier (1993). *El abecé de MS-DOS 6*. Madrid: Ra-Ma.

Ceballos, Fco. Javier (1995). *Curso de Programación en C/C++*. Madrid: Ra-Ma.

Diego García, Emilio de (1995). *Historia de la industria en España: la electrónica y la informática*. Madrid: Escuela de Organización Industrial.

Fernández Garrido, Javier (2000). *Windows 98*. Parla: AFISA.

Gil Orihuel, A.; Rieiro Marín, I. (1987). *Historia de la informática*. Madrid: Alhambra.

Kernighan, Brian W.; Ritchie, Dennis M. (1991). *El lenguaje de programación C*. México: Prentice-Hall.

Martín, José María (2003). *Hardware microinformático*. Madrid: Ra-Ma.

Nodo Tau. *Introducción a los sistemas operativos*.http://www.tau.org.ar/base/lara.pue.udlap.mx/sistoper/capitulo1.html [Consulta: Febrero de 2004]

Organik, Elliot (1978). *Programming language structures*. Nueva York: Greenwood Press.

Pedrera Carvajal, Andrés; Sánchez Figueroa, Fernando (1997). *Informática práctica*. Cáceres: Universidad de Extremadura.

Shannon, Claude E. (1981). *Teoría matemática de la comunicación*. Madrid: Forja.

Tanenbaum, Andrew (1991). *Sistemas operativos: Diseño e implementación* (1.ª ed.). Prentice Hall.

Vaquero Sánchez, Antonio (1997). *Microsoft Windows NT server: kit de recursos*. Madrid: McGraw-Hill.

Almacenamiento y bases de datos

Ramon Costa i Pujol
Jaume Raventós Moret

PID_00153114

UOC
Universitat Oberta de Catalunya
www.uoc.edu

Índice

Introducción

La aplicación práctica de los ordenadores y de los sistemas informáticos es el **tratamiento de la información**. Por eso, al conjunto de sistemas informáticos y mecanismos de comunicación se lo denomina *tecnologías de la información y la comunicación* (TIC).

Ved también

Para saber qué comporta el tratamiento de la información en un sistema informático, podéis consultar el subapartado "Los ordenadores tratan información" en el módulo "Ordenadores y sistemas operativos".

Todos los programas informáticos gestionan datos, que son captados, analizados, tratados y presentados a fin de que el usuario pueda tomar decisiones o llevar a cabo otras acciones sobre ellas.

Por lo tanto, es necesario poder **almacenar los datos** e informaciones que el ordenador y los sistemas de información tienen que tratar.

En un ordenador hay dos tipos **de memoria principal**: RAM y ROM.

Ved también

Para tener más detalles sobre el funcionamiento de la memoria RAM y ROM, podéis consultar el subapartado"Estructura básica de un ordenador" en el módulo "Ordenadores y sistemas operativos".

- La información de la **memoria ROM** no se borra nunca. Almacena el test de fiabilidad del ordenador (un conjunto de instrucciones por medio de las cuales comprueba el estado de sus componentes cada vez que se enciende), las rutinas de inicio y arranque, y la BIOS.

  Sistema básico de entrada y salida

  La BIOS (*basic input / output system*) es un pequeño conjunto de instrucciones que utiliza el procesador para encontrar el sistema operativo y cargarlo a la memoria RAM. También sirve para gestionar el flujo de datos entre el sistema operativo y los dispositivos del ordenador.

ROM

ROM (*read only memory*) es memoria de sólo lectura.

Esta memoria es **permanente**, es decir, la información no se pierde al desconectar el ordenador. Las instrucciones y los datos de la ROM permanecen allí cuando se apaga el ordenador.

- A la **memoria RAM** se guarda temporalmente la información de los programas y procesos que se ejecutan. Se puede leer y escribir. Para que un programa se pueda ejecutar se tiene que cargar (almacenar) en la memoria RAM.

RAM

RAM (*random access memory*) es memoria de acceso aleatorio.

Esta memoria es **volátil**, es decir, en el momento en que se apaga el ordenador se pierde toda la información que hay almacenada.

Estos tipos de memoria, pues, no cumplen un requisito imprescindible para almacenar datos que es el "almacenamiento permanente de gran cantidad de información", por lo que son necesarios **dispositivos de almacenamiento** externo, como, por ejemplo, los discos magnéticos, los CD-ROM, los DVD o los lápices de memoria (*pen drive* o *USB flash drive*).

Ved también

Para conocer los diferentes dispositivos de almacenamiento externo, podéis consultar el subapartado "¿Cuáles son las unidades de almacenamiento más corrientes?" del apartado "Soportes físicos de información" en el módulo "Ordenadores y sistemas operativos".

Por otra parte, esta información que hay que almacenar y gestionar se tiene que estructurar de manera que la puedan tratar fácilmente tanto los ordenadores y programas informáticos como, sobre todo, sus usuarios.

Por ello, existen diferentes **estructuras de datos** que permiten almacenar esta información: los ficheros y las bases de datos. Este módulo trata los principales aspectos relacionados con estas dos maneras de organizar la información.

- En un **fichero** se almacena información que hace referencia al mismo tema de una manera estructurada para poder trabajar con los datos de manera individual.
  Cada fichero se compone de estructuras más simples llamadas **registros**. Cada registro está formado por **campos** que contienen información en lo referente a un elemento o característica en particular dentro del fichero.
  Las principales operaciones que se pueden llevar a cabo sobre un fichero son su creación, la lectura y actualización de la información (registros y campos), la ordenación y consulta de los registros, su eliminación y la generación de informes.
  El tipo de organización de un fichero (secuencial, directa, etc.) condiciona las operaciones que se pueden hacer con él y, por lo tanto, viene determinado por la aplicación que se le dé.
  Vistas sus limitaciones, los ficheros han sido sustituidos, en la mayoría de aplicaciones, por bases de datos.

- Las **bases de datos** surgieron para intentar resolver los problemas que tenían los ficheros (inconsistencia de los datos, redundancia, rigidez en las buscas, dependencia de los programas...).

Una base de datos es una serie de datos organizados de manera que se pueda acceder a sus contenidos y éstos se puedan administrar y actualizar con facilidad.

Un **sistema gestor de bases de datos** (SGBD) es el conjunto de software destinado a la creación, gestión, control y manipulación de la información almacenada en una base de datos.

El SGBD dispone de un **lenguaje de definición de datos** (DDL) que permite definir el esquema de la base de datos, un **lenguaje de manipulación de datos** (DML) que facilita la gestión de la información almacenada, y una **interfaz de usuario** que permite acceder al sistema y trabajar con comodidad.

Existen tipos muy diversos de bases de datos. Esta asignatura se centra en las relacionales (el modelo más utilizado), que permiten gestionar información estructurada, y en las documentales, que gestionan información no estructurada.

- Una **base de datos relacional** está formada por **tablas**, **que son estructuras de filas** (los **registros** de información) y **columnas** (los **campos** de información de los registros).
  El diseño y creación de una base de datos relacional consiste en definir el **modelo relacional**, es decir, las tablas, sus campos y las relaciones entre tablas, principalmente.
  Previo a este diseño, sin embargo, se aconseja **elaborar el modelo conceptual**, que permite plasmar en un esquema entidad-relación las diferentes informaciones (entidades, atributos e interrelaciones) que se quieren almacenar y gestionar en la futura base de datos relacional.

- Una **base de datos documental** se caracteriza porque cada registro se corresponde con un documento de cualquier tipo (publicación, documento gráfico o sonoro...) o con su referencia.
  La información que contiene se estructura en diferentes campos: unos se refieren a la descripción formal del documento, otros tratan sobre su contenido temático, e incluso uno puede guardar el propio documento.
  Según la información contenida y la referencia al documento correspondiente, se puede clasificar en: base de datos de **texto completo** (contiene los propios documentos), **archivo electrónico de imágenes** (contiene imágenes de los documentos originales) y base de datos **referenciales** (contiene referencias para localizar los documentos originales).

Objetivos

Los **objetivos generales** que el estudiante puede alcanzar son los siguientes:

1. Identificar la necesidad de las bases de datos (BD), y sus ventajas en relación con los ficheros.
2. Conocer los elementos de un sistema gestor de bases de datos (SGBD) y sus principales funcionalidades.
3. Adquirir una visión general de las tipologías de BD, especialmente las relacionales y las documentales.
4. Conocer los fundamentos del diseño de BD relacionales.

Estos objetivos generales se desglosan en los **objetivos específicos** siguientes:

1. Identificar la necesidad de almacenar y gestionar la información por parte de las aplicaciones informáticas.
2. Definir los conceptos de fichero, registro, campo, registro lógico frente a registro físico y bloque.
3. Identificar el proceso de intercambio de información entre la memoria externa y la interna de un ordenador.
4. Definir las principales operaciones que se pueden llevar a cabo con un fichero.
5. Clasificar los ficheros según dos criterios: la longitud de los registros y su uso.
6. Describir las diferentes maneras como se puede organizar un fichero.
7. Enumerar las limitaciones de uso y aplicación de los ficheros.
8. Definir las principales funcionalidades de las BD.
9. Enumerar aplicaciones prácticas de las bases de datos.
10. Definir las principales funcionalidades y componentes o elementos de un SGBD.

11. Identificar diferentes SGBD existentes en el mercado.

12. Enumerar los principales tipos de BD y su manera de organizar los datos.

13. Describir los principales conceptos de las bases de datos relacionales y la estructuración de sus datos.

14. Distinguir entre los lenguajes de definición de los datos (DDL) y los de manipulación de estos datos (DML).

15. Identificar las opciones disponibles para trabajar con una BD desde un lenguaje de programación.

16. Identificar los principales elementos de un modelo conceptual de bases de datos según el modelo entidad-relación.

17. Definir una base de datos relacional a partir de un modelo entidad-relación.

18. Enumerar las diferentes opciones para normalizar el diseño de una BD.

19. Conocer los conceptos básicos de bases de datos documentales (BDD).

20. Enumerar los diferentes sistemas de recuperación de la información en bases de datos documentales.

21. Distinguir entre distintas categorías de bases de datos documentales.

22. Identificar tres tipos diferentes de modos de acceso a la información de una base de datos documental.

23. Distinguir entre las BD centradas en un solo tipo de documento y las que incorporan diferentes tipos de documentos.

24. Diferenciar las bases de datos documentales según el modelo de tratamiento documental.

1. Ficheros

Un fichero, también denominado *archivo*, es un conjunto ordenado de datos que mantienen entre sí una relación lógica y se almacenan en un soporte de información para la comunicación con el ordenador.

Ejemplo de estructura de un fichero

El fichero que almacena los datos de los abonados a plazas de aparcamiento de un garaje se puede describir gráficamente de la siguiente manera:

Nombre de abonado	NIF de abonado	Matricula de vehículo	Tipo de vehículo	Plaza de aparcamiento	....
Ramon Pujol	46587875	2587-CDD	monovolumen	25	
Jaume Moret	46585965	5874-AAF	furgoneta	36	
Marc Costa	45898745	2698-CCE	todo terreno	24	
Marta Pou	58954785	2598-BDA	motocicleta	14	
Albert Raventós	54785478	5698-BEE	motocicleta	05	
Montse Molina	58745854	1985-CCA	monovolumen	32	
Núria Camps	47858585	8847-TKM	ciclomotor	18	
....					

En un fichero se almacena información que hace referencia al mismo tema de una manera estructurada para poder trabajar con los datos de manera individual.

Los ficheros se componen de estructuras más simples denominadas *registros*. Todos los registros de un fichero son del mismo tipo.

Cada registro está formado por **campos** que contienen información en lo referente a un elemento o característica en particular dentro del fichero.

Fichero de propietarios de plazas de aparcamiento en un garaje

Gráficamente, podríamos describir un fichero de la manera siguiente:

Propietario	NIF	Matrícula	Plaza de aparcamiento	...
Ramón Pujol	46587875	2587-CDD	25	

Propietario	NIF	Matrícula	Plaza de aparcamiento	...
Alberto Costa	45898745	2698-CCE	24	
Antonio Pou	58954785	2598-BDA	14	
Marcos Cartagena	54785478	5698-BEE	05	
Montse Monte	58745854	6985-CCA	36	
Nuria Camps	47858585	B-5847-TX	14	
Pedro Soler	46585965	5874-AAF	36	
...				
...				

Dentro de un fichero determinado, los registros se identificarán por un campo o un conjunto de campos denominados *clave*. Estos identificadores servirán para distinguir cada registro de los otros, así como para facilitar la localización rápida de los registros dentro de los ficheros (por ejemplo, el NIF). Un fichero puede tener una clave, varias o ninguna.

La mayoría del software (procesadores de texto, gestores de hojas de cálculo, etc.) también trabaja con ficheros para poder almacenar la información que gestiona. Por ejemplo, cualquier documento elaborado con un procesador de texto se guarda como un fichero, así como cualquier foto o vídeo que almacenemos en nuestro disco duro. En estos casos, sin embargo, la información se visualiza por pantalla, de una manera diferente a como se ha guardado. El programa (procesador de texto, sistema operativo, etc.) se encarga de leer el contenido de aquel fichero y mostrarlo de una manera comprensible para el usuario.

Los archivos se almacenan en los dispositivos de **memoria masiva**, también denominados de *memoria auxiliar*, como por ejemplo los discos duros, los disquetes, CD, DVD, lápiz de memoria (*pen drive* o *flash memory*) y otros dispositivos de memoria con conexión USB.

Para saber más

Para conocer las características de los diferentes tipos de soportes de almacenaje, podéis consultar el subapartado 1.4.6 del módulo "Ordenadores y sistemas operativos".

Distintas denominaciones para el almacenaje permanente de datos

Hay muchas maneras de denominar un soporte físico digital de almacenaje permanente de datos, según los términos utilizados para su descripción: dispositivo (unidad o soporte) de almacenaje (memoria) auxiliar (secundario/a o externo/a o masivo/va). Así, es correcto usar, entre otros, cualquiera de las siguientes denominaciones genéricas:

- Dispositivo de almacenamiento secundario.
- Dispositivo de memoria auxiliar.
- Dispositivo de memoria masiva.
- Soporte de almacenaje externo.
- Soporte de almacenaje masivo.

- Unidad de memoria externa.
- Unidad de almacenaje secundario.
- Etc.

Un elemento que se debe considerar en los ficheros, tal y como se comentará más adelante, es su organización, de la que distinguiremos cuatro tipos: **secuencial**, **secuencial encadenada**, **secuencial indexada** y **directa o aleatoria**. Este aspecto, tal como se comenta más adelante, está relacionado con el tipo de acceso a los ficheros.

Para saber más

Para conocer cómo se organizan y cómo es el acceso a los ficheros, podéis consultar el subapartado 1.4 de este módulo.

En los siguientes apartados, también se presentan las principales operaciones que se pueden hacer con los ficheros y las diferentes tipologías en las que éstos se clasifican.

1.1. Proceso de entrada y salida de datos. Almacenaje

Un fichero está compuesto por un conjunto de registros, todos del mismo tipo (por ejemplo, los datos de los propietarios de los coches vendidos en un concesionario), y cada registro está compuesto por campos que contienen información en lo referente a un elemento o característica del registro (por ejemplo, el número de la matrícula, modelo y marca del coche, NIF del propietario, fecha de la venta, etc.).

Proceso de entrada y salida de datos

Los procesos de entrada y salida de datos corresponden a las operaciones de escritura y lectura, respectivamente, mediante las que se añade, modifica y/o se accede a la información de los registros.

El **acceso a los dispositivos** de memoria masiva donde se almacenan los ficheros puede ser de dos tipos: secuencial y directo.

- En los **soportes de acceso secuencial** (por ejemplo, una cinta) para acceder al registro *n* se tienen que leer, uno a uno, los *n-1* registros anteriores en el orden en el que están escritos.

- En los **soportes de acceso directo** (por ejemplo, un disco duro o un disquete), se puede acceder directamente a un registro físico solamente dando su dirección física.

Se diferencia entre **registro lógico**, entendido como la información correspondiente a uno de los elementos del fichero, y **registro físico** (o bloque), que es el conjunto de información que se puede leer y escribir al mismo tiempo en el soporte físico.

Los registros físicos se almacenan en el dispositivo correspondiente, y el sistema operativo es el encargado de escribir y leer los datos que componen el fichero. El sistema operativo transporta, cada vez que se accede al dispositivo (para leer o escribir), una cantidad fija de información (bloque o registro físico) que depende de las características físicas o del hardware.

Para saber más

Para saber más sobre el proceso de escritura y lectura de datos en un fichero, podéis consultar el sistema de ficheros y la gestión de E/S del sistema operativo en el subapartado 2.2 del módulo "Ordenadores y sistemas operativos".

En general, un bloque puede tener un número variable de registros lógicos, es decir, se pueden transferir distintos registros lógicos en una sola operación de escritura/lectura. Este hecho recibe el nombre de **bloqueo**. El número de registros lógicos contenidos en un bloque (longitud del bloque) recibe el nombre de **factor de bloqueo**.

El bloqueo de registro aporta dos grandes ventajas:

- Más velocidad en los procesos de entrada/salida (E/S), ya que cuanto mayor es el factor de bloqueo, menos accesos serán necesarios hacer en el dispositivo físico para tratar la información.

- Mejor aprovechamiento de la capacidad de soporte del almacenamiento.

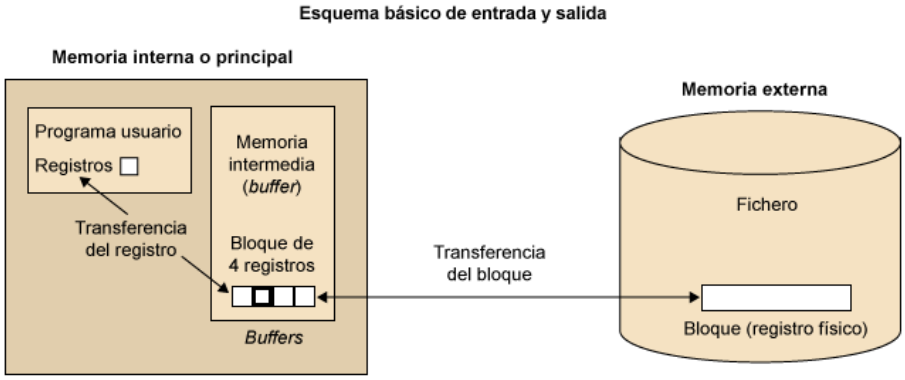
La **dirección lógica** (dirección software) de un registro es la posición relativa que ocupa en el fichero, mientras que la **dirección física** es la posición real donde se encuentra el registro en el soporte de información (dirección hardware).

En el fichero, los registros se aparecen al usuario de manera lógica, es decir, ordenados según su dirección lógica. En cambio, el orden físico de los registros de un fichero en el disco puede no tener ninguna relación con la información que contiene.

El **sistema operativo** es el responsable de realizar la transformación de **la dirección lógica**, utilizada en los programas, en **la dirección física** con la que se almacena en el dispositivo.

Desde un programa se accede a un fichero para leer y modificar uno de sus registros o escribir. Al leer, se transfiere de bloque a bloque (registro físico) la información del fichero a un área de memoria principal denominada *memoria intermedia* (*buffer*).

Desde esta memoria intermedia se transfiere la información que el programa puede programar. Y de la misma manera, el programa puede transferir información desde esta zona al fichero y modificar su contenido.



Aquí el factor de bloqueo (número de registros contenidos en un bloque) es cuatro.

Desde la memoria intermedia, los datos son procesados por el programa.

De este modo, el programa transfiere información desde esta memoria al fichero, con lo cual se actualiza su contenido.

1.2. Operaciones con ficheros

Los programas, o aplicaciones informáticas, acceden a los ficheros para llevar a cabo un conjunto de **operaciones** con la información almacenada. A continuación, se describen las operaciones más habituales:

a) Creación. Es la primera operación que se debe hacer sobre un fichero. Consiste en definir las características de los datos. A partir de este momento, ya se puede añadir registros al fichero y trabajar con ellos.

b) Lectura. Consiste en recuperar la información del fichero en el nivel del registro.

c) Mantenimiento o **actualización**. La modificación de un fichero incluye tres acciones posibles:

- **Inserción** de un registro nuevo.
- **Modificación** de un registro, cambiando uno o más valores.
- **Eliminación** de un registro.

d) Ordenación. Consiste en alterar el orden de los registros de un fichero según uno o más criterios, y a partir de los valores de uno o más campos, que reciben el nombre de claves de ordenación.

e) Búsqueda. Operación que consiste en localizar dentro de un fichero un registro concreto. Hay diferentes procedimientos para llevar a cabo la búsqueda de uno o más registros en un fichero:

- **Búsqueda secuencial**. Representa examinar cada uno de los registros del fichero hasta que se encuentra el solicitado.

- **Búsqueda dicotómica o binaria**. Aplicable sólo en el caso de ficheros ordenados. Consiste en dividir los ficheros en subficheros cada vez más y más pequeños. Se empieza leyendo el registro del medio y, si el valor que se ha de buscar es superior, se descartan los registros de la primera mitad del fichero. Si el valor que se busca es inferior, se descartan los de la segunda mitad del fichero. Con los registros elegidos se repite el proceso de búsqueda, hasta encontrar el registro buscado o comprobar que no está.

- **Búsqueda por bloques**. Con este método se realiza el mismo recorrido sobre las claves que en un fichero secuencialmente indexado. En primer lugar, se determinará el bloque en el que se encuentra el registro. Para hacerlo, se lee el último registro de cada bloque, hasta encontrar el buscado o uno mayor. En este último caso, se pasará a buscar el registro en el bloque anterior y se efectúa una búsqueda secuencial.

Para saber más

Para tener más información sobre la ordenación de ficheros, podéis consultar el apartado "Organización de ficheros" en este módulo.

f) Fusión. En esta operación se unen dos o más ficheros para integrarlos en uno solo.

g) División. Operación contraria a la fusión y que divide la información de un fichero en dos o más.

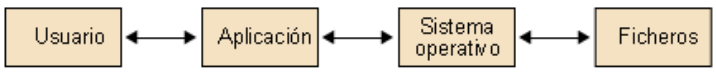
h) Generación de informes a partir de datos contenidos en el fichero. Los informes pueden ser impresos, por pantalla, en formato web, etc.

i) Destrucción. Eliminación del fichero del dispositivo correspondiente.

Las aplicaciones informáticas y los sistemas operativos incluyen una serie de programas útiles para llevar a cabo las operaciones básicas con ficheros: creación, eliminación, lectura, etc. Estos programas reciben el nombre de **sistemas de gestión de ficheros**.

En general, un fichero utilizado por un usuario desde un lenguaje de alto nivel, por ejemplo Visual Basic, C++ o Java, no lo gestiona directamente el mismo programa, sino el sistema operativo, responsable de realizar los accesos necesarios al dispositivo en el que se archiva y de transferir la información solicitada del fichero al programa y a la inversa.

Gestión del sistema operativo



1.3. Tipos de ficheros

Los ficheros se pueden clasificar según dos criterios principales:

- Según la longitud de los registros que contienen. En este caso, se distingue entre los ficheros de longitud fija, los de longitud variable, los delimitados y los indefinidos.

- Según el uso que se hará de éstos. Se pueden clasificar entre ficheros permanentes y temporales, y en cada uno de estos casos se distinguen entre ficheros maestros o de situación, ficheros constantes o ficheros históricos e intermedios, de maniobras o de resultados.

A continuación, se describen las características principales de cada uno de estos tipos de ficheros.

1.3.1. Según la longitud de los registros

Los registros que forman parte de un fichero pueden tener la misma longitud o una diferente, tanto si se debe a la existencia de campos de longitud variable como al hecho de que contienen campos de longitud fija pero que se repiten un número variable de veces, así como si es por las dos causas.

Si se sigue este criterio, los ficheros se pueden clasificar o distinguir entre los tipos siguientes:

- **Ficheros de longitud fija**. Son aquéllos en los que la suma de los caracteres de todos los campos de los registros es constante. Todos los registros, por lo tanto, tienen la misma longitud.

Ficheros de longitud fija

R	A	M	O	N					C	O	S	T	A					3	6	5	4	-	C	D	A		
P	E	D	R	O					P	U	J	O	L					B	-	2	5	4	5	-	T	X	
...																											
...																											
...																											

- **Ficheros de longitud variable**. Son los ficheros en los que cada registro puede tener una longitud diferente en el fichero, que oscila entre una pequeña variación entre un mínimo y un máximo. En estos casos, el sistema reserva una palabra al inicio de cada registro, en la que anota su longitud.

Ficheros de longitud variable

#18#	R	A	M	O	N	C	O	S	T	A	3	6	5	4	-	C	D	A	#19	P	E	D	R	O
P	U	J	O	L	S	B	-	2	5	4	5	-	T	X	#	-	-	-	-		-	-	-	-
-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-

El número que acompaña al símbolo # indica la longitud de cada registro.

- **Ficheros delimitados**. En estos tipos de ficheros, la longitud del registro es variable y no se puede saber en qué medida difieren unos de otros. El sistema incluye un carácter especial para indicar el final del registro. En estos casos, se dice que los ficheros son de tipo texto.

Ficheros delimitados

R	A	M	O	N	C	O	S	T	A	3	6	5	4	-	C	D	A	;	P	E	R	E	P
U	J	O	L	S	B	-	2	5	4	5	-	T	X	;	-	-	-	-	-		-	-	-
-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-

En este caso, el caracter punto y coma (";") indica dónde acaba cada registro.

- **Ficheros indefinidos**. Son aquéllos en los que la longitud es totalmente variable. En estos casos, el sistema operativo no realiza ninguna gestión sobre la longitud de los registros del fichero. Será el programa el encargado de localizar el principio y el final de cada registro.

Ficheros indefinidos

R	A	M	O	N	C	O	S	T	A	3	6	5	4	-	C	D	A	P	E	D	R	O	P	U
J	O	L	S	B	-	2	5	4	5	-	T	X	-	-	-	-	-	-	-		-	-	-	-
-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-

1.3.2. Según el uso

Según el uso que se hace de un fichero, los podemos clasificar en diferentes tipos. Para poder organizar y diseñar un fichero, es imprescindible saber qué función ejercerá.

a) Ficheros permanentes. Contienen la información necesaria para el funcionamiento de una aplicación. Tienen una vida larga y normalmente no se pueden generar de manera inmediata a partir de otros ficheros. Se distinguen tres tipos:

- **Ficheros maestros o de situación**. Contienen información que refleja el estado actual de los datos. Se actualizan periódicamente para adaptarse a cada situación nueva. Los registros se modifican muy a menudo, pero la estructura no varía. La mayoría de las aplicaciones informáticas están orientadas a actualizar el fichero maestro o a obtener resultados a partir de este fichero.

Ejemplo de ficheros maestros

Un ejemplo de fichero maestro sería los datos de los clientes de una cadena de supermercados que tienen su tarjeta de fidelización.

- **Ficheros constantes**. Mantienen datos fijos para la aplicación. Estos datos son prácticamente inamovibles (tienen pocas modificaciones) y se utilizan como ficheros de consulta.

Ejemplo ficheros constantes

Un fichero que guarde la relación de los códigos de identificación de categorías de productos sería un ejemplo de fichero con datos constantes.

- **Ficheros históricos**. Contienen datos que fueron actuales en tiempos anteriores. Están constituidos por registros que reúnen cronológicamente las diferentes modificaciones que ha experimentado el fichero en el tiempo. Se conservan con el objetivo de poder reconstruir situaciones anteriores. Algunas veces pueden estar formados, simplemente, por los registros borrados del fichero maestro.

Ejemplo ficheros históricos

Un ejemplo de este tipo de ficheros sería la lista de todos los clientes que se han dado de baja de la cadena de supermercados que tenían su tarjeta de fidelización.

b) Ficheros temporales. Estos ficheros contienen información necesaria para un proceso específico dentro de una aplicación. Se generan a partir de los datos de los ficheros permanentes y se utilizan para obtener resultados o actualizar la información. Una vez hecha la operación, se eliminan. Se distinguen tres tipos:

- **Ficheros intermedios**. Generados a partir de los resultados de un programa y utilizados como la entrada en otro dentro de la misma tarea o proceso. Sólo se utilizan para pasar información de un proceso a otro.

Ejemplo ficheros intermedios

Un ejemplo de éstos sería un programa que necesitara hacer una operación de cálculo con los pedidos de todos los clientes de un área geográfica concreta. Se podría generar el fichero con estos datos a partir del fichero maestro, llevar a cabo el tratamiento sobre este fichero intermedio y eliminarlo posteriormente.

- **Ficheros de maniobras**. Este tipo de ficheros se utiliza para no perder la información generada por un proceso que no se puede conservar, por falta de espacio, en la memoria principal.

Ejemplo ficheros de maniobras

Un ejemplo de aplicación de un fichero de maniobras sería el de un proceso de una entidad financiera que quisiera calcular todos los intereses que ha cobrado a todos los clientes durante los últimos tres años.

- **Ficheros de resultados**. Se generan a partir de los resultados finales de un proceso, que se transferirán a un dispositivo de salida.

Ejemplo ficheros de resultados

Un ejemplo de este tipo sería un fichero de impresión con el listado de todos los clientes del supermercado que han hecho compras superiores a un importe determinado durante los dos meses anteriores, y que después se transferirá a la impresora.

1.4. Organización de ficheros

El ordenador debe tener acceso a los ficheros creados por los usuarios, tanto para leer la información almacenada como para grabarla o modificarla.

El **acceso a un fichero** está íntimamente relacionado con su **organización**. Esta organización indica cómo están dispuestos los registros en el **soporte material** para conseguir una utilización más eficiente.

Para saber más

Para tener más información sobre los registros en el soporte material, podéis consultar el apartado "Soportes físicos de información" en el módulo "Ordenadores y sistemas operativos".

Al crear un fichero, se especifica cuál será su organización, dado que determinará el tipo de acceso que podremos utilizar.

Hay **dos tipos de acceso** a un fichero:

- **Acceso secuencial**. Para acceder al registro *n*, primero debemos recurrir a los *n – 1* anteriores en el mismo orden en el que se escribieron, hasta encontrar el registro buscado.

- **Acceso relativo**. A partir de una clave (identificador), se puede acceder directamente al registro sin tener que recurrir a los anteriores.

Clave de un fichero

La clave (o identificador) de un fichero es un campo o un conjunto de campos que identifican, de manera única, cada uno de los registros del mismo. Un ejemplo de clave es el NIF, tal como se muestra en el ejemplo de estructura de un fichero que podéis ver al principio del apartado "Ficheros".

Los tipos de organización de ficheros son básicamente cuatro: **secuencial**, **secuencial encadenada**, **secuencial indexada** y **directa**. En este apartado, se describe cada uno de estos cuatro tipos de organización.

1.4.1. Organización secuencial

En este primer tipo de organización, los registros se almacenan de manera **continua**, uno a continuación del otro, sin espacio entre los mismos y sin ningún índice que indique sus direcciones.

Ésta es la única organización que se puede gestionar en un dispositivo no direccionable como, por ejemplo, las cintas magnéticas, lo cual no implica que no se pueda utilizar en soportes de acceso directo.

Esta organización se suele utilizar con ficheros en los que en cada proceso se debe acceder a la mayor parte de los registros. Tienen la **ventaja** de que aprovechan muy bien el espacio, son sencillos de utilizar y se pueden utilizar con dispositivos secuenciales.

Su principal **inconveniente** es la falta de flexibilidad y la lenta velocidad de acceso, por lo cual no se recomiendan para ser utilizados en procesos interactivos.

En este tipo de ficheros se pueden realizar las operaciones de añadir registros al final del fichero y de consultarlos. En cambio, para insertar, modificar o borrar registros se deben utilizar ficheros temporales.

Ejemplo de organización secuencial

Un ejemplo de aplicación de esta organización sería un fichero que recogiera y almacenara, en tiempo real, las condiciones ambientales (temperatura, humedad, etc.) a inter-

valos de diez segundos. Posteriormente, esta información se podría tratar si se quisiera generar alguna media.

1.4.2. Organización secuencial encadenada

Los registros de un fichero con esta organización almacenan, además de su información, un **puntero** con la dirección del registro siguiente, según el orden lógico del fichero. Desde este punto de vista lógico, el fichero se ordenará según el valor de una clave, si bien los registros se colocan en direcciones físicas totalmente arbitrarias. Los punteros garantizan la secuencia lógica del fichero.

Puntero de un registro

Un puntero (o apuntador) es una variable cuyo valor es la dirección física donde se almacena un dato. Se dice que la variable *p* de tipo puntero (que contiene la dirección donde se almacena el valor *v*) apunta a *v*. Esta marca interna que tiene un registro siempre apunta al registro lógico activo (es decir, al registro lógico que se procesa en la siguiente operación del fichero) y se incrementa automáticamente cada vez que se procesa un registro (se lee o se escribe).

Esta organización es adecuada en ficheros que utilizan procesos interactivos con una actualización frecuente, pero en los que cada operación afecta a pocos registros.

Presentan la **ventaja** de una gran flexibilidad, ya que se pueden realizar todo tipo de operaciones (añadir, consultar, insertar, modificar y borrar), pero en cambio tienen el **inconveniente**, igual que los ficheros con organización secuencial para acceder a un registro concreto, de que hay que seguir de manera secuencial, todos los registros según los punteros.

Ejemplo de organización secuencial encadenada

Un fichero de ejemplo con una organización secuencial encadenada sería la lista de los socios de una entidad deportiva de volumen medio en la que no haya que llevar a cabo muchas operaciones sobre estos datos.

1.4.3. Organización secuencial indexada

Un fichero con organización secuencial indexada está compuesto de dos zonas:

- La zona de **registros**, que contiene todos los registros ordenados según el valor de alguna clave (uno o más campos del registro que la identifican). Se puede considerar una estructura secuencial pura.

- La zona de **índices**, formada por un número de registros inferior al total de registros del fichero.

Los registros de la zona de índices tienen una estructura particular que no tiene nada que ver con los registros reales del fichero. Tienen el **campo clave** (que contiene algunos valores de la clave del fichero) y el **campo dirección** (que contiene la dirección de un registro del fichero).

La zona de registros se considera dividida en una serie de **tramos lógicos** o **segmentos**, cada uno formado por registros consecutivos. Para cada tramo en la zona de registros, hay un registro en la zona de índices que contendrá en su campo clave el valor de la clave del último registro del tramo y, en el campo dirección, la dirección del primer registro del tramo.

Ejemplo de fichero con organización secuencial indexada

Ejemplo de fichero con organización secuencial indexada

Zona de índices	
Clave	Dirección
34874644	1
45674635	4
47657462	7
59263527	10

Zona de registros				
Núm. reg.	NIF	Nombre	...	
1	32564736			Segmento I
2	34563546			
3	34874644			
4	44756575			Segmento II
5	45673653			
6	45674635			
7	45674651			Segmento III
8	46587689			
9	47657462			
10	56734521			Segmento IV
11	58374632			
12	59263527			

En la zona de índices, el **campo clave** corresponde al NIF, que es la clave del fichero en la zona de registros, y el **campo dirección** corresponde al número de registro del fichero en la zona de registros.

Una analogía de esta organización es el índice de un libro de lectura.

La **ventaja** de este tipo de organización es que resulta muy útil cuando hay que combinar consultas a registros concretos y el procesamiento secuencial de todo el fichero, ya que permite el acceso directo a los registros en determinadas aplicaciones y, secuencialmente, en otras.

Su principal **inconveniente** es la imposibilidad de introducir nuevos registros o actualizaciones en los registros que hay en el fichero sin tener que llevar a cabo una reorganización con las modificaciones efectuadas.

Una solución a este problema es reservar una zona de memoria complementaria, donde se almacenen los registros nuevos y que se suele denominar *zona de desbordamiento* o *ciclo de trabajo*.

Una mejora para esta estructura consiste en utilizar la organización denominada *secuencial indexada encadenada* (una mezcla de secuencial indexada y secuencial encadenada).

Ejemplo de organización secuencial indexada

Una aplicación de esta estructura organizativa sería la de un fichero que contuviera todos los datos de las reservas de productos de un almacén, consultado de manera muy frecuente, pero que se actualiza una vez cada semana.

1.4.4. Organización directa o aleatoria

En este tipo de organización no hay ninguna relación lógica entre los registros y su ubicación física.

Cada registro se sitúa en una dirección de memoria que se calcula para cada uno aplicando una fórmula o algoritmo matemático. Estos métodos toman el valor de un campo del registro, aplican una transformación y obtienen su dirección.

Cuando se detecta que la dirección asignada a un registro ya está ocupada por otro, se puede optar por dos alternativas:

- Buscar una nueva dirección libre, secuencialmente, en el mismo fichero hasta encontrar una **posición libre** donde almacenar los registros. Este proceso es lento y provoca una mala ocupación de la memoria (degrada el fichero), pues quedan espacios vacíos.

- Reservar una **zona de desbordamiento** en la que almacenar, de manera consecutiva, los sinónimos (registros que obtienen el mismo valor de la dirección, y por lo tanto, pueden ocupar una misma posición), a medida que éstos aparecen.

Nota

- **Colisión**: situación que se produce cuando se quiere poner un registro en una posición del fichero que ya está ocupada por otro registro.
- **Sinónimos**: registros que entran en colisión ya que, por transformación, obtienen un mismo valor de la posición o dirección física.
- **Zona de desbordamiento**: zona donde se almacenan los registros sinónimos que no se pueden colocar en la posición que les corresponde de la zona de registros por estar ocupada por otro registro. También se denomina área de excedentes, zona de saturación u *overflow*.

Los ficheros con esta organización tienen como principales **ventajas** la gran flexibilidad y la rapidez de consulta.

Entre los **inconvenientes**, hay que destacar el desperdicio del espacio (ya que se debe reservar todo el espacio necesario, aunque no haya datos) y la necesidad de utilizar soportes direccionables.

Ejemplo de organización directa o aleatoria

El fichero con los datos de los vehículos que tienen contratada una plaza en un aparcamiento privado de 2.000 plazas. Como no se puede guardar un registro para cada matrícula posible, habrá que "generar" la posición del registro en función, por ejemplo, de los números de la matrícula. Una posible fórmula (o algoritmo de conversión) es coger las cuatro cifras de la matrícula; si la primera cifra está entre 0 y 4, ésta se cambia por un 0 y si está entre 5 y 9, se cambia por un 1. Esta nueva combinación de cifras indicará la posición en la que se almacenará el registro en el fichero. Además, se reserva espacio para 200 registros adicionales, por ejemplo, en una zona de desbordamiento al final del registro.

En este ejemplo, una posible distribución sería la siguiente:

Núm. registro	Matrícula	Propietario	Núm. plaza	...
...				
0233	T-3233-BC			
...				
0323	L-3323-CD			

Núm. registro	Matrícula	Propietario	Núm. plaza	...
...				
0333	T-4333-BB			
...				
0367	G-3367-AQ			
...				
0421	T-3421-EF			
...				
0454	B-3454-ZY			
...				
0456	B-3456-SZ			
...				
0543	L-4543-AA			
...				
0565	L-4565-AB			
...				
0567	T-4567-AB			
...				
0576	B-4576-VU			
...				
0587	G-4587-AW			
...				
1458	B-5458-TY			
...				
1675	G-8675-DD			
1676	G-5676-ED			
...				
1786	T-9786-DX			
1787	B-6787-VB			
...				
Zona de ciclo de trabajo				
2000	T-3454-CC			

Núm. registro	Matrícula	Propietario	Núm. plaza	...
2001	L-9786-DD			
...				
2199	...			

Los dos registros almacenados en la zona de desbordamiento no se pueden ubicar en la zona normal de registros, dado que su posición natural está ocupada. Son sinónimos de otros registros que ya ocupan el espacio que les correspondería a ellos.

1.5. Limitaciones de los ficheros

Aunque en su momento los ficheros fueron un elemento importante para el almacenamiento de la información (y de hecho, todavía lo son como soporte de muchas aplicaciones informáticas), su uso y trabajo supone, sin embargo, una serie de inconvenientes, y los principales son:

- **Inconsistencia de la información**. Al tener información parcial o totalmente duplicada en varios ficheros, o al tener organizaciones diferentes, la actualización de los datos puede ser costosa. Un error en la actualización de esta información puede provocar inconsistencias en los datos del fichero.

Ejemplo de inconsistencia de la información

Por ejemplo, si queremos almacenar la información de los pacientes de un hospital con su médico asociado, y también los datos de los médicos, habrá que disponer de dos ficheros. Parte de la información de los médicos estará repetida en los dos ficheros.

- **Redundancia**. Este problema ocurre al tener datos que no aportan información y que se pueden calcular a partir de otros datos.

Ejemplo de redundancia

Muchas veces, con el objetivo de facilitar la rapidez del acceso a la información, por ejemplo, se pueden guardar datos en el fichero que se podrían calcular a partir de otros campos almacenados. Por ejemplo, el importe con IVA de las facturas del fichero de facturas de proveedores de la empresa.

- **Rigidez de busca**. Cada fichero tiene una organización determinada, según el tipo de acceso para el cual se ha definido.

Ejemplo rigidez de busca

Si se ha creado un fichero de tipo secuencial, por ejemplo, siempre se deberá acceder a él de esta manera, aunque su contenido pueda evolucionar y se pueda aplicar otro método de busca.

- **Dependencia de los programas**. Las relaciones entre los datos no se almacenan a su lado. El hecho de conocer y mantener estas relaciones es responsabilidad del programa que las gestiona. Cualquier cambio en la estructura del fichero representa modificar los programas que lo utilizan.

Ejemplo dependencia de los programas

Los programas informáticos desarrollados para gestionar la información han de conocer la estructura y la disposición de los campos de los ficheros que tratan. Añadir, por ejemplo, un nuevo campo al fichero de los pacientes del hospital significaría tener que modificar todos los programas que trabajan con este fichero para que puedan consultar la información de este nuevo campo.

Por este motivo, en la mayoría de las aplicaciones informáticas el uso de los ficheros se ha sustituido por el trabajo con bases de datos.

1.6. Resumen

Un fichero es un conjunto ordenado de datos que mantienen una relación lógica unos con otros y se almacenan en un soporte de información para que los puedan utilizar un programa o una aplicación.

En un fichero, se almacena información que hace referencia al mismo tema de una manera estructurada para poder trabajar con los datos de modo individual.

Los ficheros están compuestos por estructuras más simples denominadas *registros*.

Cada registro está formado por **campos** que contienen información en lo referente a un elemento o característica particular dentro del fichero.

Las principales operaciones que se pueden llevar a cabo sobre un fichero son su creación, la lectura, el mantenimiento y la actualización de la información (registros y campos), la ordenación de los registros, la búsqueda o consulta de datos, la fusión de dos o más ficheros en uno solo, la división de un fichero en uno o más, su destrucción o eliminación y la generación de informes o impresos a partir de la información contenida.

Los archivos se almacenan en los **dispositivos de memoria masiva**, también denominados **de memoria auxiliar**.

Estos dispositivos pueden ser de dos tipos: secuencial y de acceso directo.

- En los **soportes secuenciales**, por ejemplo una cinta, para acceder al registro n se deben leer los $n - 1$ registros anteriores.

- En cambio, en los **soportes de acceso directo**, por ejemplo un disco duro o un disquete, se puede acceder directamente a un registro físico sólo dando la dirección física.

Los ficheros se pueden clasificar según dos criterios principales:

- Según la **longitud de los registros** que contiene. En este caso, se distingue entre los ficheros de longitud fija, los de longitud variable, los delimitados y los indefinidos.

- Según el **uso que se hará de éstos**. Se pueden clasificar entre ficheros permanentes y temporales. En el primer tipo distinguimos entre ficheros

maestros o de situación, ficheros constantes o ficheros históricos; el segundo tipo pueden ser intermedios, de maniobras o de resultados.

Los tipos de organización de ficheros son básicamente cuatro: **secuencial**, **secuencial encadenada**, **secuencial indexada** y **directa**. Su organización condicionará qué tipos de operaciones pueden hacer con el fichero. En función de la aplicación del fichero, será más aconsejable una organización u otra.

Finalmente, debemos remarcar que vistas las limitaciones de los ficheros (inconsistencia, redundancia, rigidez en las búsquedas, dependencia de los programas...), se han sustituido, en la mayoría de las aplicaciones, por bases de datos.

2. Bases de datos y sistemas gestores de bases de datos

Las bases de datos surgieron para intentar resolver los problemas que tenían los ficheros.

Una base de datos es un sistema formado por un conjunto de datos organizados de manera que se evitan los datos redundantes, son independientes de los programas que los utilizan, almacenan los datos junto con las relaciones entre ellos y se puede acceder a éstos de diferentes maneras.

Dicho de otra manera, una **base de datos (BD)** es una colección de datos organizados de manera que se pueda acceder a sus contenidos, administrarlos y actualizarlos con facilidad.

El **sistema gestor de bases de datos (SGBD)** es el conjunto de software destinado a la creación, la gestión, el control y la manipulación de la información almacenada en una base de datos.

Acrónimos más comunes en bases de datos

En este módulo, se utilizan los siguientes acrónimos, de una manera amplia usados en catalán (y en castellano):

Concepto	Acrónimo	Comentario
Base de datos	**BD**	En inglés, *DB* por *data base*
Sistema gestor de bases de datos	**SGBD**	En inglés, *DBMS* por *data base management system*
Lenguaje de definición de datos	**DDL**	Del inglés, *data definition language*
Lenguaje de manipulación de datos	**DML**	Del inglés, *data management language*
Lenguaje de consulta estructurado	**SQL**	Del inglés, *structured query language*

Los SGBD disponen de un lenguaje de definición de datos **(DDL)** que permite definir el esquema de la base de datos, un lenguaje de manipulación de datos **(DML)** que facilita la gestión de la información almacenada y una interfaz de usuario que permite el acceso y trabajo en el sistema de una manera cómoda.

Cronológicamente, las bases de datos se han clasificado en tres grupos: **jerárquicas**, **en red** y **relacionales**.

Hacia los años setenta, aparecieron las **bases de datos** (BD) relacionales para obtener más flexibilidad en el tratamiento de los datos y son, hoy día, las más extendidas.

Ejemplo de gestor de BD

Algunos ejemplos de gestores de BD relacionales serían el programa Microsoft Access, el servidor de BD Oracle, Ms SQL Server, MySQL, etc.

Una **BD relacional** está formada por tablas, que son una estructura de filas (los registros de información) y columnas (los campos de información de los registros).

El diseño y la creación de una base de datos relacional consistirá en definir el **modelo relacional**, es decir, las tablas, sus campos y las relaciones entre tablas, principalmente.

Previamente al diseño de este modelo relacional, sin embargo, se aconseja hacer el **modelo conceptual**, que permite "plasmar" en un modelo de entidad-relación las diferentes informaciones que se quieren llegar a almacenar y gestionar con la futura base de datos relacional.

Un tipo de bases de datos especial son las documentales **(BDD)**, caracterizadas por el hecho de que cada registro se corresponde con un documento de cualquier tipo (publicación, documento gráfico o sonoro, etc.) o su referencia.

Las BDD se pueden clasificar en distintas categorías, según la información que contienen y la referencia al documento correspondiente: bases de datos de texto completo, archivos electrónicos de imagen y bases de datos referenciales. También se pueden clasificar por otras tipologías: según el modo de acceso a la información, la cobertura documental o según el modelo de tratamiento documental.

En los próximos apartados, se detallará cada uno de estos aspectos.

2.1. Conceptos básicos

En los siguientes subapartados se exponen algunos conceptos básicos sobre las bases de datos y los ficheros y las bases de datos.

2.1.1. Bases de datos

Una **base de datos** es una colección de información almacenada de manera organizada en formato electrónico.

Un **sistema gestor de bases** de datos es un programa que permite organizar su almacenamiento y facilitar su recuperación.

Las bases de datos ofrecen distintas **ventajas**:

- Facilitan el almacenamiento de grandes cantidades de información.
- Facilitan la recuperación rápida y flexible de información.
- Facilitan la organización y reorganización de la información.
- Facilitan la impresión y distribución de información de diferentes maneras.

Ejemplos de bases de datos

Las bases de datos son una colección de información de cualquier tipo como, por ejemplo, un directorio telefónico, un tarjetero de recetas, un catálogo de fichas bibliográficas, el inventario de productos y servicios de la compañía, los registros de calificaciones escolares de un estudiante, la lista de las asignaturas de un curso, la relación de los trabajadores de la organización, la lista de los clientes de la compañía, los pacientes de un centro asistencial, la lista de las lecturas de la temperatura de un termómetro, etc.

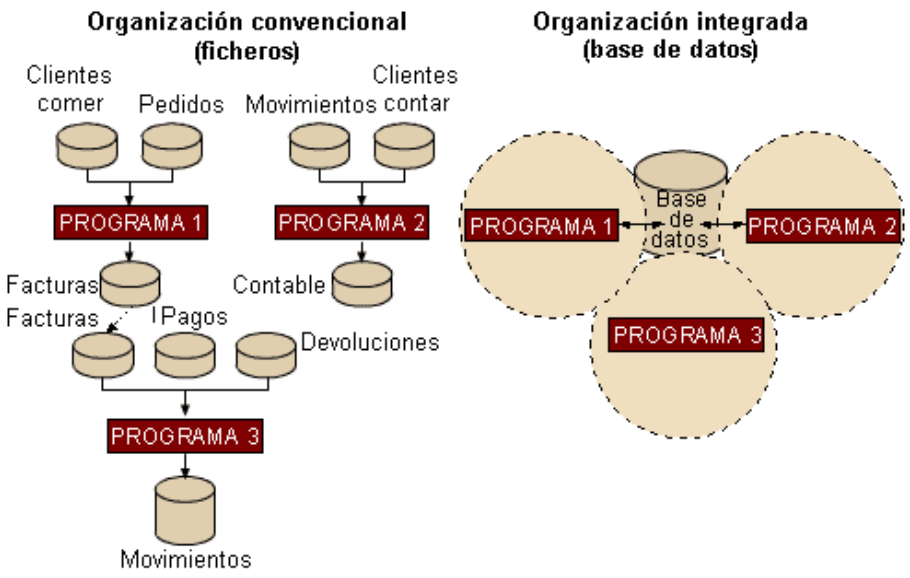
La información de algunos de estos ejemplos se puede gestionar en forma de simples ficheros, ya que contienen un único tipo de entidad (clientes, pacientes, temperaturas, calificaciones, etc.), mientras que otros integran información de diferentes tipos de entidad que sólo puede ser tratada como bases de datos.

Prácticamente, cualquier colección de información se puede convertir en una base de datos.

2.1.2. Ficheros y bases de datos

Las bases de datos aparecieron como una alternativa a los sistemas de ficheros para solucionar los inconvenientes y desventajas que presentaban.

La figura siguiente muestra la diferente forma de organizar los datos en un sistema de ficheros y en una base de datos. El primero precisa múltiples ficheros para organizar la misma información que la segunda integra en un único archivo.



Para saber más

Para tener más información sobre los inconvenientes y las desventajas que presentaban los ficheros, podéis consultar el apartado "Limitaciones de los ficheros" en este mismo módulo.

Actualmente, el uso de ficheros se limita básicamente a aplicaciones en las que no es necesario un acceso directo a la información o hay pocos tipos de datos relacionados entre diferentes ficheros, y como soporte para almacenar la información de programas (por ejemplo, procesadores de textos) o del sistema operativo.

Aun así, físicamente, en el ordenador, las bases de datos se almacenan y gestionan como ficheros por medio de los sistemas gestores de bases de datos (SGBD).

Aunque de manera muy simplificada, se pueden identificar las diferencias siguientes entre los ficheros tradicionales y las bases de datos:

- **Entidades tipo**. Los ficheros tienen registros de una sola entidad tipo (por ejemplo, pacientes), mientras que las BD tienen datos de distintas entidades tipo (pacientes, tratamientos, médicos, etc.).

  El concepto de entidad

  Tanto los ficheros como las BD almacenan información de *objetos* del mundo real, denominados "entidades", que tienen una función importante en las organizaciones. Se pueden distinguir dos niveles de abstracción de una entidad:

  - **Entidad instancia:** es un objeto real o abstracto con existencia independiente y características que lo diferencian de otros objetos, aunque sean del mismo tipo. Por ejemplo, un paciente concreto, un médico concreto, un alumno concreto, una enfermedad concreta o una asignatura concreta. En este módulo, también se usa el término **ocurrencia** para referirse a este concepto.

  - **Entidad tipo:** es un conjunto de entidades instancia que comparten las mismas características. Por ejemplo, la abstracción *paciente*, formada por todos los pacientes de un hospital, o la abstracción *alumno*, formada por todos los alumnos de una escuela. En este módulo, también se usa el término **entidad** (a secas) para referirse a este concepto.

- **Interrelaciones**. En el caso de los ficheros, el sistema no interrelaciona ficheros, mientras que en el caso de las BD el sistema tiene herramientas previstas que interrelacionan entidades.

- **Redundancia**. Mientras se crean ficheros a la medida de cada aplicación con todos los datos necesarios –aunque algunos sean redundantes con respecto a otros ficheros–, en las BD todas las aplicaciones trabajan con la misma BD y la integración de los datos es básica, de manera que se evita la redundancia.

- **Usuarios**. Los ficheros sirven para un solo usuario o una sola aplicación, mientras que las BD son compartidas por muchos usuarios de diferentes tipos.

2.2. Aplicaciones prácticas de BD

En la actualidad, la mayoría de los sistemas de información de una organización están basados en bases de datos.

Ejemplos de aplicaciones prácticas de BD

- Por ejemplo, toda la información gestionada por el departamento comercial (clientes, productos, ventas, etc.) estará en bases de datos.
- También lo estará toda la información tratada por el departamento de marketing y comunicación (campañas, actuaciones, etc.).
- Toda la información relativa a los empleados y gestionada por el Departamento de Recursos Humanos también estará en una o más bases de datos, así como toda la información utilizada por los departamentos de finanzas y compras.
- Todos los datos de un sistema de gestión de alumnos de un centro de formación, por ejemplo, también estarán en una base de datos.
- Otro ejemplo de BD sería toda la información sobre las películas de un videoclub, sus clientes y sus actuaciones (alquileres, devoluciones, etc.).
- Todas las bibliotecas basan su gestión sobre bases de datos.

Y, de esta manera, encontraríamos infinidad de ejemplos.

2.2.1. Características y objetivos de las BD

Tal y como hemos comentado en el apartado anterior, las bases de datos surgen para resolver los problemas que presentan los ficheros: inconsistencia de la información, redundancia, rigidez de busca y dependencia de los programas.

Una base de datos es un sistema formado por un conjunto de datos organizados de manera que se controla el almacenamiento de los que son redundantes. Los datos resultan independientes de los programas que los utilizan; se almacenan de manera conjunta los datos y las relaciones entre sí, por lo que se puede acceder a la información de distintas maneras.

Los requerimientos principales que debe cumplir un buen sistema de bases de datos (BD) son los siguientes:

- Distintos usuarios pueden acceder simultáneamente a la base de datos, cada uno a una información determinada.
- Se controla el acceso de los usuarios a los datos y se garantiza la confidencialidad y seguridad.
- Los datos se almacenan sin redundancia (excepto en algunos casos en los que interesa mejorar los accesos y el rendimiento).

- Permiten utilizar diferentes métodos de acceso y asegurar flexibilidad en las investigaciones.
- Disponen de mecanismos de recuperación de información.
- Se puede cambiar el apoyo físico de la base de datos sin que se resientan la base de datos ni los programas que la utilizan.
- La modificación del contenido y las relaciones entre los datos no afecta a los programas que las utilizan.
- Disponen de una interfaz de usuario que permite utilizar la base de datos de manera cómoda y flexible.
- Toda base de datos se gestiona por medio de un sistema de gestión de bases de datos (SGBD).

2.3. Sistemas gestores de BD

Se denomina *sistema gestor de base de datos* (DBMS o SGBD) al conjunto de software destinado a la creación, la gestión, el control y la manipulación de la información sobre una base de datos. Los SGBD tienen como finalidad registrar y mantener información.

Un SGBD debe permitir llevar a cabo las operaciones siguientes:

a) Definición del esquema de la base de datos. Una vez diseñado el esquema de la base de datos, lo hemos de describir mediante un conjunto de instrucciones. Esto se lleva a cabo mediante un lenguaje específico, denominado *lenguaje de descripción de datos* (DDL), y que, como cualquier lenguaje de alto nivel, necesitará un traductor para generar el código objeto a partir del código fuente.

Para saber más

Para tener más información sobre el esquema de la base de datos, podéis consultar el apartado "Definición conceptual de BD" en este mismo módulo.

b) Acceso a la información desde un lenguaje de alto nivel. Esto se realiza mediante un lenguaje específico, denominado *lenguaje de manipulación de datos* (DML). El DML se puede utilizar de dos maneras diferentes:

Para saber más

Para saber más sobre los lenguajes de definición (DDL) y de manipulación (DML) de datos, podéis consultar el apartado "Lenguajes de bases de datos relacionales", en este mismo módulo.

- Como lenguaje huésped, incluyendo sentencias DML dentro de un programa anfitrión escrito en un lenguaje de alto nivel (como, por ejemplo, Cobol, Pascal, Basic, C, etc.).
- De manera interactiva, por medio de programas que contengan exclusivamente sentencias propias del DML.

c) Acceso a la información en modo conversacional. El SGBD incorpora una interfaz de usuario por medio de la cual el usuario puede introducir sentencias DDL o DML directamente desde un terminal y obtener información interactiva.

d) Gestión de archivos. Función realizada por un módulo denominado *gestor de archivos* que se encarga de la comunicación con el sistema operativo.

e) Otras funciones: controles de usuarios, recuperación de la información después de errores del sistema, organización física de la base de datos, control de seguridad y privacidad de información o la gestión de accesos concurrentes.

Para saber más

Para tener más información sobre sistemas operativos, podéis consultar la unidad "Sistemas operativos" del módulo "Ordenadores y sistemas operativos".

En un sistema tradicional de ficheros, cada una de las aplicaciones deberá realizar la descripción de los registros, así como determinar la organización de ficheros, los tipos de acceso, etc.

Para saber más

Para tener más información sobre ficheros, podéis consultar la unidad "Ficheros" de este mismo módulo.

En un entorno de base de datos, estas especificaciones las realizan los SGBD y es el administrador de la base de datos el encargado de garantizar que se efectúen éstas y otras funciones sobre el sistema.

2.3.1. Características y objetivos de los SGBD

A continuación, se detallan algunas de estas características o requisitos de los sistemas gestores de bases de datos (SGBD):

- **Consultas no predefinidas y complejas**. El objetivo fundamental de los SGBD es permitir que se hagan consultas no predefinidas (*ad hoc*) y complejas. El usuario debe poder formular la consulta con un lenguaje sencillo, que se quede en el nivel lógico, y el sistema lo debe interpretar directamente. Los usuarios podrán hacer consultas de cualquier tipo y complejidad directamente al SGBD, que deberá responder inmediatamente sin que estén preestablecidas, es decir, sin que se deba escribir, compilar y ejecutar un programa específico para cada consulta. En cambio, en los ficheros tradicionales, cada vez que se quería realizar una consulta se debía escribir un programa a medida.

- **Flexibilidad e independencia**. La complejidad de las BD y la necesidad de ir adaptándolas a la evolución del sistema de información hacen que un objetivo básico de los SGBD sea dar flexibilidad a los cambios. Interesa obtener la máxima independencia posible entre los datos y los procesos y programas para que se puedan efectuar todo tipo de cambios tecnológicos y variaciones en la descripción de la BD, sin que se deban modificar los programas de aplicación ya escritos ni se haya de cambiar la manera de escribir las consultas directas. En el mundo de los ficheros ya había independencia física en un grado determinado, pero en el mundo de las BD suele ser mucho mayor.

- **Mínima redundancia**. Uno de los objetivos de los SGBD es facilitar la eliminación de la redundancia, ya que suponía un riesgo de inconsistencia o incoherencia de los datos. Conviene hacer que un dato figure una sola vez en la BD. En los ficheros tradicionales, cada aplicación utilizaba su fichero. Sin embargo, puesto que había mucha coincidencia de datos entre aplicaciones, se producía mucha redundancia entre los ficheros.

- **Concurrencia de usuarios**. Un objetivo fundamental de los SGBD es permitir que diferentes usuarios puedan acceder de manera concurrente a la misma BD. Con el objetivo de tratar los accesos concurrentes, los SGBD utilizan el concepto de transacción de BD, concepto de especial utilidad para todo aquello que hace referencia a la integridad de los datos. Se denomina *transacción de BD* o simplemente *transacción* a un conjunto de operaciones simples que se ejecutan como una unidad. Los SGBD han de conseguir que el conjunto de operaciones de una transacción nunca se ejecute parcialmente. O se ejecutan todas, o no se ejecuta ninguna.

- **Mecanismos de seguridad**. En el campo de los SGBD, el término *seguridad* se suele utilizar para hacer referencia a los temas relativos a la confidencialidad, las identificaciones, las autorizaciones, los derechos de acceso, etc. Los SGBD permiten definir autorizaciones o derechos de acceso en diferentes ámbitos (globalmente en toda la BD, en la entidad y en el atributo). Estos mecanismos de seguridad requieren que el usuario se pueda identificar. Se suelen utilizar códigos de usuario (y grupos de usuarios) acompañados de contraseñas (*passwords*), pero también se utilizan tarjetas magnéticas, identificación por reconocimiento de la voz, etc.

2.3.2. SGBD del mercado

Hoy día, hay muchos paquetes en el mercado que reducen costes y esfuerzos y hacen de las bases de datos una aplicación práctica para la mayoría de los usuarios de ordenador.

Estos paquetes son gestores de bases de datos sofisticados y poderosos que tratan de satisfacer las necesidades de dos grupos de usuarios:

- **Usuarios finales**, que son los que quieren utilizar la base de datos para almacenar, organizar y recuperar los datos.

Ejemplo de SGBD para usuarios finales

Entre los SGBD para el entorno Windows en nivel de usuario final, destaca Microsoft Access.

- **Programadores**, que la quieren utilizar para desarrollar aplicaciones específicas para los negocios u organizaciones.

Ejemplo de SGBD para programadores

Entre los SGBD servidores, pensados para departamentos de organizaciones y empresas, destacan los productos comerciales Informix, Microsoft SQL Server y el servidor Oracle. Los principales SGBD de software libre son MySQL y PostgreSQL.

Entre los paquetes líderes para Windows, en un ámbito de usuario destaca Microsoft Access.

Entre los sistemas gestores de bases de datos servidores, es decir, los pensados para departamentos de organizaciones y empresas, destacan los servidores Oracle, Microsoft SQL Server, Informix, MySQL, etc.

2.4. Tipos de bases de datos

Si atendemos al modelo lógico de datos, hay varios tipos de BD: **jerárquicas, en red, relacionales y orientados a objetos**, o combinaciones entre diferentes tipos. Por ejemplo, hoy en día Oracle es un SGBD para bases de datos relacionales y orientados a objetos.

Para saber más

Para saber más sobre modelos de datos y sus elementos constitutivos, podéis consultar el subapartado "Modelos de datos: conceptos básicos".

Modelo de datos

Un **modelo de datos** es, por una parte, la descripción del contenedor de datos (donde se guardará la información) y por la otra, el conjunto de métodos para almacenar y recuperar información de los contenedores.

Todo modelo de datos proporciona una serie de elementos (objetos, asociaciones, propiedades, operaciones, restricciones, etc.) que permiten, mediante la abstracción de una parte del mundo real, la obtención de un conjunto estructurado de datos y un conjunto de operaciones definidas sobre ellas.

A continuación, se detallan los tipos de BD atendiendo a su **modelo de datos**:

1) BD jerárquica

El primer modelo de BD en aparecer fue el **modelo jerárquico**, a principios de los años sesenta. Tal como indica su nombre, almacena la información en una estructura jerárquica.

Los datos se organizan en registros interrelacionados en forma similar a un árbol invertido. Un árbol se compone de un conjunto de registros (*nodos*) unidos por medio de asociaciones (*arcos*) padre-hijo que pueden ser de uno a uno o de uno a muchos. Es decir, un nodo "padre" puede tener uno o más nodos "hijos". El nodo que no tiene padres se denomina *raíz*, y los nodos que no tienen hijos, *hojas*. Estas asociaciones entre nodos se implementan, físicamente, utilizando punteros.

Para saber más

Podéis encontrar una pequeña descripción de cómo funciona el puntero de un registro en el subapartado "Organización secuencial encadenada".

El término *nodo*, cuando hablamos de bases de datos, tiene las siguientes acepciones:

- En una BD jerárquica o en red, **cada uno de los objetos** (entidades o registros) que conforman la estructura de datos (en árbol o en red, respectivamente).

- En una BD distribuida, **cada uno de los ordenadores** o lugares interconectados donde se emplazan los datos.

Ejemplos de SGBD jerárquicos

El IMS (*information management system*) de IBM, diseñado en 1966 y aparecido en 1968, fue líder de los SGBD jerárquicos durante los años setenta. A pesar de ser superado por otros modelos de datos, es el único que ha sobrevivido, pues desarrollos posteriores han permitido que soporte aplicaciones en Java, JDBC, XML y servicios web.

Otros SGBD de la época, actualmente en desuso, son System-2000 (de MRI, después comercializado por SAS Institute), Mark IV (de Control Data Corporation) y TDMS (de System Development Corporation).

El modelo jerárquico y las estructuras en árbol se utilizan en otros sistemas de almacenamiento, como el registro de Windows, los documentos XML o los repositorios LDAP (basados en el protocolo de acceso a servicios de directorio).

Su principal **inconveniente** es la incapacidad de representar la redundancia de datos (duplicidad de registros) y la poca flexibilidad.

2) BD en red

A finales de los años sesenta, aparecieron SGBD basados en un **modelo en red**. Como en el modelo jerárquico, hay registros e interrelaciones, pero un registro (*nodo*) ya no está limitado a ser "hijo" de un solo "padre". Un nodo puede tener diferentes padres, permitiendo así representar interrelaciones más complejas que con el modelo jerárquico.

Tiene la ventaja de que permite representar cualquier sistema y disponer de datos redundantes, pero su administración es muy compleja, por lo que su uso se ha limitado siempre a los programadores y no a los usuarios finales.

El comité CODASYL-DBTG propuso, en 1971, un estándar basado en este modelo, que fue adoptado por muchos productores de SGBD; sin embargo, encontró la oposición de IBM, la empresa dominante por entonces.

Ejemplos de SGBD basados en el modelo en red

El primero en aparecer fue el IDS (*integrated data store*), desarrollado por Charles W. Bachman a mitad de los años sesenta para la compañía General Electric. Versiones posteriores, como el IDS/2, fueron comercializadas por Honeywell-Bull.

El SGBD comercial más destacado fue el IDMS (*integrated database management system*) de Cullinet (actualmente, de Computer Associates), a pesar de que no cumple plenamente el estándar CODASYL.

Otros sistemas conocidos incluyen VAX-DBMS (de Digital), IMAGE (de Hewlet-Packard) y DMS_1100 (de Univac).

Tanto los sistemas del modelo en red como del jerárquico –que se puede considerar un caso particular del primero– presentaban lenguajes procedimentales que obligaban al programador a navegar, registro a registro, por la BD, no disponiendo de suficiente independencia físico-lógica, lo cual comportaba escasa flexibilidad.

3) BD relacional

La introducción de la teoría matemática del álgebra relacional, en el campo de las BD por parte de Edgar F. Codd (de IBM), en 1970, le llevó a proponer el **modelo relacional**.

Para saber más

Para tener más detalles sobre el modelo relacional, podéis consultar el subapartado siguiente, "Bases de datos relacionales".

Se basa en el concepto matemático de relación (que tiene una apariencia similar a una tabla de valores) y en la teoría de conjuntos. Una **tabla** es una estructura bidimensional donde se almacenan los datos como un conjunto de registros del mismo tipo, que se dividen horizontalmente en filas y verticalmente en columnas. Una **fila** representa un registro o grupo de valores de campos, y una **columna** contiene información en lo referente a un único campo o atributo de diferentes registros.

Este modelo supuso un paso importante para el desarrollo de los SGBD. Durante los años ochenta, se extendió el uso de los sistemas relacionales, la mayoría de los cuales utilizan como lenguaje nativo para la manipulación de los datos el SQL (estandarizado en 1986).

Para saber más

Para saber más sobre el SQL, podéis consultar el subapartado "Lenguajes de bases de datos relacionales".

Ejemplos de SGBDR

El prototipo System R (de IBM) fue el origen de los primeros SGBD relacionales como Ingres (creado en 1974 por la Universidad de Berkeley, y comercializado en 1980 por Ingres Inc. y después por Computer Associates).

Hay centenares de SGBD relacionales, tanto para PC (monousuario) como para sistemas *mainframe* (multiusuario), a pesar de que muchos no son completamente fieles al modelo relacional.

Los primeros SGBD comerciales multiusuario fueron Oracle (1979) y DB2 de IBM (1982). Otros sistemas multiusuario son Informix (después Informix Dynamic Server, de IBM), Sybase (SQL Server, después Adaptive Server Enterprise), Microsoft SQL Server, Allbase (de Hewlet-Packard), Interbase (de Borland) y los SGBD de código abierto MySQL, Firebird (basado en Interbase) y OpenIngres (de Computer Associates).

Los SGBD relacionales para microordenadores, inicialmente monousuario, después ofrecieron arquitectura cliente-servidor y se han ido adaptando al estándar ODBC (*open database connectivity*) de Microsoft, que permite la utilización de herramientas de tipo *front-end*. Son ejemplos de ello: Paradox y dBase (de Borland), FoxPro y R:base (de Microverso) y Access de Microsoft.

La mayoría de sistemas informáticos hoy en funcionamiento utilizan un SGBD relacional (o extensiones del mismo), aunque en algunas organizaciones todavía se usan los jerárquicos.

4) BD de objetos u orientada a objetos

Estructura la información en clases y subclases de objetos completos (estado y comportamiento) e incorpora los conceptos importantes de la programación orientada a objetos (encapsulación, herencia, polimorfismo, etc.). Todos los objetos pertenecen a una clase de objetos generalizados que comparten las mismas funciones. Por medio de la herencia, los objetos de una clase pueden adquirir las funciones de la clase.

Los siguientes conceptos del enfoque orientado a objetos son importantes para los sistemas de BD:

- **Objeto**: entidad discreta que tiene, básicamente, dos componentes:
  - Estado (valor) o estructura con sus atributos.
  - Comportamiento, que queda determinado por las **operaciones** (procedimientos, métodos y funciones) que se le pueden aplicar externamente.

- **Clase**: categoría generalizada que describe las características y el comportamiento de un grupo de objetos que pueden existir dentro de la misma.

- **Herencia**: transferencia de las propiedades y el comportamiento de una clase existente a una nueva subclase que se deriva de ella.

- **Método**: implementación de una operación que se aplica externamente a los objetos.

- **Mensaje**: código de programa que se envía al objeto para llamar o invocar un método.

- **Encapsulación**: acción de ocultar la estructura y el comportamiento interno de un objeto detallando con rigor su comportamiento externo, cosa que permite impedir conflictos y accesos incorrectos.

- **Polimorfismo**: capacidad para que varias clases de objetos tengan un comportamiento diferente ante una misma operación.

Correspondencia entre los términos de OO y los de programación tradicional

		OO	Programación
Términos	Objeto		Variable
	Clase		Tipo
	Método		Procedimiento, rutina (secuencia de instrucciones que ejecutan una única función)
	Mensaje		Llamamiento a rutina o procedimiento

Ejemplos de SGBDO

Destacan: GemStone (de ServiLogic), ObjectStore (de Object Design), Ardent (antes O2), Objectivity, Versant, Ontos y Poet.

Gestores de objetos

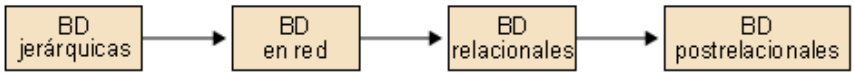
Sistemas con características similares a los SGBDO, pero con un modelo de datos limitado, que son extensiones de sistemas de ficheros o de gestión de memoria virtual. Por ejemplo, Mneme (gestor de memorias de traducción) y LOOM (sistema gestor de bases del conocimiento).

El modelo orientado al objeto surgió a finales de los años ochenta, por la falta de capacidad semántica del modelo relacional a la hora de atender determinados tipos de aplicaciones que requieren modelar objetos e interrelaciones complejos, almacenar información no estructurada, etc.

Esto ha hecho que los conceptos de orientación a objetos (OO) se hayan incorporado a los sistemas relacionales.

Tradicionalmente, las BD se han clasificado de modo cronológico en generaciones:

- 1.ª generación: **BD prerrelacionales**: jerárquicas y en red.
- 2.ª generación: **BD relacionales**.
- 3.ª generación: **BD postrelacionales**: orientadas al objeto, relacionales extendidas, distribuidas, documentales, etc.



Evolución cronológica de las bases de datos.

Las **BD postrelacionales** incluyen muchas tipologías y combinaciones de BD, algunas de las cuales no responden a un modelo de datos definido. A continuación, se exponen las más destacadas:

1) BD relacionales extendidas

Sistemas aparecidos, a partir de los años ochenta, fruto de la evolución del modelo relacional para superar sus limitaciones y obtener una mayor capacidad expresiva que represente más fielmente el mundo real, incorporando funcionalidades de otros modelos de datos como orientación a objetos, deductivas, con mecanismos de actividad, etc.

Se pueden distinguir diferentes enfoques:

a) BD relacional con frontal OO: añade un nivel o capa de OO superpuesta a un SGBD relacional preexistente.

b) BD objeto-relacional (SGBDOR): sigue el enfoque integrador de los dos modelos, incorporando nuevas capacidades:

- Nuevos tipos de datos (definidos por el usuario, multimedia, objetos grandes BLOB) que permiten gestionar aplicaciones más complejas con una gran riqueza de dominios.

- Operaciones que permiten gestionar el comportamiento de los tipos de datos.

- Mayor capacidad expresiva para la semántica de los datos (disparadores, procedimientos almacenados y funciones definidas por el usuario).

- Reusabilidad de librerías de clases ya existentes.

- Mejor capacidad consultiva (consultas anidadas, recursivas, almacenadas, prefabricadas, etc.).

En realidad, los actuales SGBD objeto-relacionales suelen incorporar capacidades deductivas, mecanismos de actividad, integración de documentos XML, etc.

c) BD activa: permite definir "reglas activas" que se activan, automáticamente, cuando suceden determinados "acontecimientos", que inician la ejecución de una "acción" (disparador o *trigger*) si se dan unas "condiciones" específicas.

Permite modelar mejor las reglas de funcionamiento interno de una organización: mantener automáticamente datos derivados, notificar determinadas situaciones, garantizar el cumplimiento de restricciones de integridad (como las reglas de negocio en aplicaciones organizativas complejas), etc.

d) BD deductiva: incluye mecanismos de inferencia (basados en "reglas deductivas" de la lógica matemática) que permiten generar información adicional a partir de los hechos almacenados en la BD. Se relaciona con la inteligencia artificial y las bases de conocimiento.

Permite la comprobación de hipótesis, el descubrimiento de conocimiento deductivo (nuevas relaciones entre los datos), la gestión de procesos gobernados por reglas, el modelado de la estructura y los procesos de la empresa, el establecimiento de perfiles de clientes en comercio electrónico, etc.

Los intentos de proporcionar un modelo de datos que represente más fielmente el mundo real han dado lugar a los **modelos de datos semánticos**.

Extensión del modelo relacional de Codd

A la hora de intentar enmendar algunas de las deficiencias de su modelo relacional, Codd presentó una versión extendida denominada RM/T (1979) y, más recientemente, RM/V2 (1990).

El *RM/T* aborda algunos aspectos similares al modelo entidad-relación (de Chen, 1976) pero de manera más cuidada. Por ejemplo, clasifica las entidades en tres categorías (núcleos, características y asociaciones), los aspectos estructurales y de integridad son más amplios y están definidos con más precisión, incluye operadores especiales adicionales, incorpora soporte para la dimensión tiempo y para distintas clases de agregación de datos.

Ejemplos de SGBDR extendidos

- **BD relacionales con frontal OO**: evoluciones de SGBDR preexistentes son Oracle 8, Informix 9, DB2/UBD (*universal database*, de IBM) y Sybase. En la década del 2000, aparecen sistemas de código abierto como OpenODB (de Hewlet-Packard, basado en el SGBDR Allbase/SQL) y PostgreSQL (basado en Ingres).

- **BD objeto-relacionales** (SGBDOR): en 1990, aparece UniSQL/X, el primer **SGBDO con SQL** partiendo de cero. A mediados de los años noventa, aparecen Illustra (después adquirido por Informix y presentado como Informix Universal Server) y Omniscience.

- **BD activas:** muchos SGBD relacionales actuales (Oracle, DB2, Sybase) cuentan con esta funcionalidad, que proporcionan en forma de disparadores. Starburst es un prototipo experimental.

- **BD deductivas:** a mediados de los años ochenta, aparece LDL (*logic data language*) y los sistemas experimentales Coral y Nail.

- A pesar de no ser extensiones relacionales, hay que mencionar las BDOO con extensión deductiva (denominadas BDDOO) creadas a finales de los años ochenta, como Validity (de Hervor) y Coral++.

2) BD temporal

Soporta la variable tiempo y otros conceptos temporales. Permite almacenar un historial de cambios y consultar el estado (actual o pasado) de la BD. En algunos casos, incluso, se almacena información futura prevista. La información temporal se puede referir a hechos puntuales (que se producen en un punto del tiempo) o de duración (en la que se considera que son verdaderos).

Aplicaciones de las BD temporales

- **Asistencia médica**, donde hay que guardar los historiales médicos de los pacientes.

- **Seguros**, donde se necesitan los historiales de reclamaciones y notificaciones de accidentes, así como información sobre los periodos de vigencia de las pólizas.

- **Reservas** (compañías aéreas, servicios de transporte, alquiler de coches, hostelería, espectáculos, etc.), donde hace falta información sobre fechas y periodos de validez de las reservas.

- **Recursos humanos**, donde hay que mantener los historiales laborales de los trabajadores de una empresa (referentes a los salarios, puestos de trabajo y proyectos en los que han trabajado).

- **Investigación científica**, donde hay que guardar los valores y periodos de tiempo en los que se miden los datos recogidos en los experimentos.

- **Gestión académica**, donde hay que incluir el semestre y/o el año de las calificaciones de las asignaturas en expedientes académicos de los alumnos y en información relativa a becas.

3) BD espacial

Proporciona conceptos que permiten seguir la pista de objetos que se encuentran en un espacio multidimensional. Facilita interpretar y especificar las características espaciales de los objetos mediante conceptos geométricos bidimensionales y tridimensionales, y gestionar las relaciones espaciales entre los objetos.

Los diferentes tipos de bases de datos espaciales son los siguientes:

- **BD cartográficas**, que almacenan datos de mapas. Incluyen conceptos geométricos bidimensionales (puntos, líneas, arcos, polígonos, etc.) y tienen en cuenta operaciones espaciales (cálculo de distancias) y operaciones booleanas (verificación de superposiciones). Se usan en gestión medioambiental, de emergencias, de conflictos bélicos.

- **BD meteorológicas**, que almacenan datos de temperaturas y otra información del tiempo atmosférico relacionada con puntos espaciales tridimensionales.

4) Sistema de información geográfica (SIG)

Permite almacenar, analizar y representar gráficamente información que describe diferentes propiedades geográficas y es gestionada por una BD, la cual contiene dos tipos de datos:

- Datos espaciales: de tipo físico (orografía, topografía y meteorología), de fronteras políticas y administrativas, de redes de transporte y comunicaciones.
- Datos no espaciales: demográficas, económicas, comerciales.

Los sistemas de información geográfica cubren disciplinas tan distintas como cartografía animada, mapeo interactivo, herramientas de ayuda a la decisión espacial, estándares de intercambio de datos geográficos, control de calidad y reingeniería, BD espacio-temporales, servicios de localización. Se pueden agrupar en tres categorías:

Ejemplos de SIG

Suelen ser sistemas que operan con SGBD relacionales o de objetos. ARC/Info, aparecido en 1981, integra la funcionalidad de un SGBD relacional. ARC/Storm (*arc store manager*) maneja BD distribuidas y se integra con SGBD relacionales como Oracle, Informix y Sybase.

- **Aplicaciones cartográficas**: estudios del terreno y el paisaje, análisis estructurales de tráfico y redes de transporte, prospección marina y petrolera, control y optimización de plantaciones agrícolas. Representan los datos mediante atributos espaciales (como la densidad de cultivos) y comportan funciones como la superposición de varias capas de mapas para combinar atributos que permitan la medida de distancias en un espacio tridimensional.

- **Aplicaciones para el modelado del terreno**: análisis del sol, gestión de recursos del agua, control de inundaciones, estudios de contaminación e impacto ambiental. Representan los datos mediante atributos espaciales (como las características del sol o la calidad del aire) y requieren la representación digital de elevaciones del terreno en puntos de muestreo que se interconectan por interpolación, dando lugar a un modelo de la superficie en 3D.

- **Aplicaciones de objetos geográficos**: análisis geográfico de mercados, análisis de pautas de voto, distribución y consumo de servicios públicos, sistemas de navegación de vehículos, gestión del catastro y localización de mobiliario urbano. Representan los objetos de interés (como distritos electorales, edificios, semáforos, farolas, centrales eléctricas) y necesitan funciones espaciales adicionales para manejar los datos referentes a un dominio físico concreto como redes viarias, de telecomunicaciones, de suministro, y de recursos hídricos; es decir, carreteras, cables de fibra óptica y de alta tensión, conductas de gas y de agua, ríos, etc.

5) BD multidimensional

Organiza los datos en estructuras matriciales de varias dimensiones y dispone de lenguajes especiales para realizar consultas complejas mediante el acceso multidimensional a los datos, de forma más adecuada y eficiente que una BD estructurada convencional como la relacional (orientada a transacciones y de naturaleza volátil).

Esta disposición proporciona una forma más natural de análisis de información, pues facilita el acceso y visualización de datos en cualquier combinación de dimensiones (las dimensiones pueden ser, por ejemplo, los productos, las zonas comerciales o los periodos impositivos de una empresa) mediante diferentes criterios y jerarquías (niveles de agrupamiento dentro de las dimensiones, subtotales previamente calculados). Representaciones jerárquicas como la exploración ascendente y la exploración descendente permiten agrupar (o resumir) los datos en categorías más generales (por ejemplo, ventas trimestrales para familias de producto) o disgregarlas en categorías más concretas.

Se implementa sobre un motor relacional con tablas, donde se almacena un volumen de datos muy elevado, a los que se aplican operaciones de cálculo intensivo (agregación, suma, media) y técnicas de indexación especiales que permiten consultas rápidas predefinidas.

Procesamiento de datos en línea: tecnologías OLAP y OLTP

A veces, las BD **multidimensionales** se denominan BD de **procesamiento analítico en línea** (OLAP, *on-line analytical processing*), ya que utilizan esta tecnología de análisis de datos que considera diferentes dimensiones y variables al mismo tiempo.

Los denominados **almacenes de datos** (*data warehouse*) utilizan toda la potencia de procesamiento analítico en línea de las BD multidimensionales.

Las BD convencionales (como las **relacionales**) soportan **procesamiento de transacciones en línea** (OLTP, *on-line transaction processing*), puesto que están optimizadas para procesar un gran volumen de consultas y transacciones concurrentes (inserciones, actualizaciones y supresiones de algunos registros), que pueden abarcar una pequeña parte de la BD, con una frecuencia elevada y una alta velocidad de respuesta.

Empresas del sector servicios (hoteles, bancos, líneas aéreas, compañías de seguros, empresas de servicio público y de comunicaciones) lo utilizan como sistema operacional de consultas y/o reservas durante 24 horas al día, 7 días a la semana.

La información se puede organizar en varias dimensiones:

a) Tabla relacional (una dimensión). Estructura los datos en registros (filas) formados por campos (columnas) de información relacionada de una única entidad u objeto del mundo real; es decir, de una sola dimensión (y una variable). A menudo, sin embargo, interesa gestionar más de una dimensión en una misma tabla.

Ejemplo de tabla relacional

La tabla siguiente muestra información de una única dimensión, la correspondiente a las zonas comerciales de una empresa (zona) con la variable *ventas*.

ZONA	Distribuidor	Dirección	Teléfono	*Ventas*
Zona1	Norte	Pl. Cataluña	93.345.89.78	103.600
Zona2	Sur	C/ Marina	93.123.98.78	250.204
Zona3	Centro	Av. Riera	93.333.88.77	303.667
Zona4	Oeste	C/ Mayor	93.123.55.65	238.423
...				

En este caso, puede interesar que haya otra dimensión (por ejemplo, producto) para almacenar las ventas de cada uno de los productos de la empresa, tal como muestra el ejemplo siguiente, de matriz bidimensional.

b) Matriz de datos bidimensional (dos dimensiones). Permite almacenar, conjuntamente, información de dos dimensiones; es decir, integra datos de dos entidades.

Ejemplo de matriz de datos bidimensional

Una hoja de cálculo con las ventas realizadas en cada zona, para cada producto, durante un periodo de tiempo determinado, ejemplifica una matriz de datos bidimensional. Las zonas se pueden mostrar en filas, y los productos, en columnas; los datos de las ventas de cada producto en cada zona están contenidos en las celdas. En este caso, hay dos dimensiones (zona y producto) y una variable (ventas). Esta es la forma multidimensional de almacenar los datos.

Ventas	**Producto**				
ZONA		**Producto1**	**Producto2**	**Producto3**	**...**
	Zona1	120.000	188.050	123.000	
	Zona2	111.000	229.005	128.976	
	Zona3	230.400	234.000	90.000	
	Zona4	162.050	222.000	89.640	
	...				

A continuación, se muestra un ejemplo de matriz bidimensional con **datos totalizados**.

Ventas	**Producto**					
Zona		**Producto1**	**Producto2**	**Producto3**	**...**	**Total**
	Zona1	120.000	188.050	123.000		**431.050**
	Zona2	111.000	229.005	128.976		**468.981**
	Zona3	230.400	234.000	90.000		**554.400**

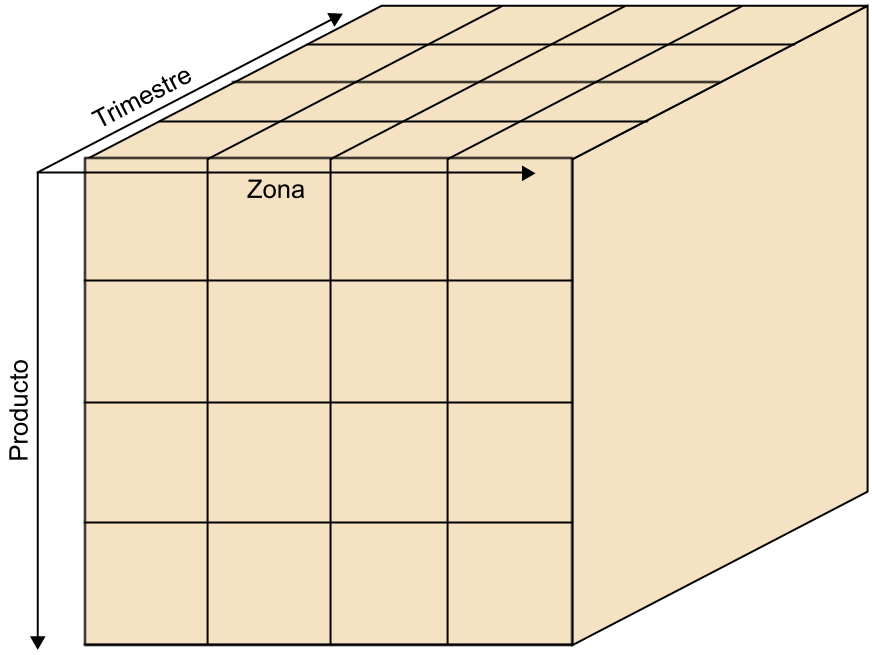
Ventas	Producto					
	Zona4	162.050	222.000	89.640		**473.690**
	...					
	Total	**623.450**	**873.055**	**431.616**		**1.928.121**

Así es como se mostrarían estos datos en un informe de un **almacén de datos** (*data warehouse*).

c) Cubo de datos (tres dimensiones). Corresponde a una matriz de datos tridimensional.

Ejemplo de cubo de datos

Si se añade una tercera dimensión temporal como los trimestres impositivos, se obtiene un cubo de datos que organiza las ventas de productos por trimestres impositivos y zonas comerciales. Cada celda contiene los datos de ventas de un producto concreto, un trimestre concreto y una zona concreta.



El cambio de orientación de un cubo de datos se consigue mediante la rotación (pivotación) sobre uno de los ejes. Por ejemplo, la rotación sobre el eje de zonas permitiría mostrar las ventas por zona en las filas, las ventas por trimestre en las columnas, y las ventas por producto en la tercera dimensión.

d) Hipercubo de datos (más de tres dimensiones). Corresponde a una matriz de datos n-dimensional ($n > 3$) a pesar de que no se puede representar gráficamente ni visualizar de forma sencilla.

6) BD paralela

SGBD que aprovecha el paralelismo de unidades de proceso o de memoria para el procesamiento paralelo de un número significativo de tareas de consulta y actualización, y de funciones de servicio asociadas (registro de transacciones, manejo de entrada/salida y almacenaje en búfer de datos) mediante la utilización concurrente de dos o más procesadores y/o dispositivos de almacenamiento. Eso proporciona mejoras significativas en el tiempo de transacción y de respuesta en BD de grandes dimensiones.

Procesamiento paralelo

Es la descomposición de las unidades de procesamiento (instrucciones individuales, secuencias de instrucciones o bloques de proceso) en partes que el sistema operativo o el SGBD asignan, para su ejecución simultánea, a procesadores diferentes que trabajan coordinadamente en un ordenador con multiprocesadores.

7) BD distribuida

Los datos están repartidos en diferentes ordenadores (nodos) de una red informática. La gran ventaja de este modelo es que permite unir información de diferentes localizaciones y acceder a ella sin tenerlo todo centralizado en un único ordenador. A pesar de la descentralización de los datos, soportada en una arquitectura básica de tipo cliente-servidor, tiene el inconveniente de que su gestión es complicada.

El desarrollo de entornos de Internet ha potenciado el acceso universal a BD distribuidas a la Web desde cualquier ordenador por medio de páginas web. Las **BD web** permiten el acceso a BD distribuidas y heterogéneas almacenadas en servidores web en archivos compartidos. Una capa de software intermedia (*middleware*) entre el usuario y el SGBD, denominada pasarela web o interfaz de pasarela común (CGI*, common gateway interface*) permite ejecutar programas externos para obtener la información que, una vez procesada, vuelve al servidor en formato HTML, que es enviada de nuevo al navegador web en el que puede ser visualizada.

Muchos SGBD actuales incorporan prestaciones para facilitar el acceso a BD por medio de Internet. Hay dos métodos principales:

- **Acceso mediante *scripts* CGI. Permite al servidor de la BD relacionarse con el servidor web mediante la especificación CGI.**

- **Acceso mediante JDBC** (*java data base connectivity*). Esta interfaz de programación de aplicaciones (API) permite a los programas en lenguaje Java el acceso a BD relacionales mediante la ejecución de consultas o sentencias SQL. Otra posibilidad es la utilización del estándar de Microsoft ODBC (*open database connectivity*).

Para saber más

Podéis ver una definición de ODBC cuando se habla de BD accesibles por Internet en el subapartado "Tipos de bases de datos".

Los SGBD que dan acceso a BD vía web proporcionan las siguientes **ventajas**:

- Permiten un acceso sencillo, flexible, homogéneo, eficiente, barato y seguro a los datos.

- Permiten el acceso a datos actualizados continuamente.

- Dan soporte a muy distintos tipos de datos difíciles y costosos de recoger por otros medios. Por ejemplo, datos multimedia como imagen, audio y vídeo.

- Evitan los problemas de compatibilidad de formatos y sistemas, facilitando el acceso desde prácticamente cualquier plataforma.

- Proporcionan herramientas de busca, indexación y relación que mejoran las capacidades y potencialidades de las BD.

8) BD XML

Sistema aparecido a principios del siglo XXI por la necesidad de almacenar y recuperar documentos XML (que son estructuras de datos semiestructurados). Hay dos enfoques de SGBD que soportan con eficiencia documentos XML:

- **Extensiones de BD para XML**. Desglosan un documento XML y lo integran en su correspondiente modelo de base de datos (relacional o de objetos).

- **SGBD XML nativos**. Respetan la estructura del documento, permiten hacer consultas sobre ésta y recuperan el documento tal como fue guardado originalmente.

Ejemplos de SGBD XML

a) Extensiones de SGBD para XML:

- SGBDR para XML: Microsoft SQLXML y Oracle XML DB son productos comerciales.
- SGBDOO para XML: Ozone/XML es de código abierto.

b) SGBD XML nativos: Tamino y X-Hive/DB son comerciales, mientras dbXML, eXist y Xindice son de código abierto.

9) BD multimedia

Ofrece características que permiten almacenar y consultar información multimedia que incluye imágenes (fotografías, dibujos, etc.), vídeo (videoclips, vídeos domésticos, películas, noticiarios o programas de televisión), audio (canciones, audiolibros, discursos o mensajes telefónicos) y documentos de texto (libros, artículos, etc.).

Para la localización de fuentes multimedia, se usan consultas de recuperación basada en contenido.

Ejemplo de recuperación basada en contenido

En una BD de vídeos, puede interesar localizar todos los vídeos que contengan una determinada persona o un tipo de actividad o acontecimiento como, por ejemplo, riadas, incendios forestales o los goles marcados por un jugador o un equipo.

Las BD multimedia abarcan disciplinas muy variadas: gestión de documentos y registros, educación y difusión de conocimientos, comunicación y entretenimiento, control de actividades en tiempo real. Se pueden clasificar en:

- **Aplicaciones de almacenaje** de gran cantidad de datos multimedia (imágenes de satélite, fotografías del espacio) mediante almacenes centrales organizados en niveles.

- **Aplicaciones de presentación** de vídeo o de audio en tiempo real, que se ven o se escuchan a medida que son enviados por el SGBD.

- **Aplicaciones de trabajo colaborativo** para analizar información multimedia en tiempo real. Se usan en campos como la telemedicina o la ingeniería.

Ejemplos de SGBD multimedia

La mayoría son SGBDO con soporte multimedia: Informix Dynamic Server, DB2 Universal Database UDB) de IBM, Oracle 8 o superior, Sybase. Otros sistemas permiten la recuperación de imágenes basada en el contenido: QBIC (*query by image content*) de IBM, Virage, Excalibur, etc.

10) BD documental

Almacena referencias y/o texto completo de documentos de propósito general (libros, artículos, cartas, contratos, etc.), normalmente sin formato predefinido, que pueden contener texto extenso y datos multimedia. Estos documentos se indexan identificando palabras clave significativas (que aparecen en el texto) y sus frecuencias relativas. Las consultas se realizan gracias a estas palabras clave o por medio de un *thesaurus* de estructura jerarquizada. A veces, se llama BD textual o sistema de búsqueda documental.

Según si incluyen o no el contenido completo de los documentos descritos, las BD documentales se clasifican en:

a) BD referencial. No contiene los documentos originales, sino información descriptiva y referencias que permiten localizarlos en otro servicio. También puede incluir enlaces para obtenerlos por medio de otro programa.

b) BD factual. Almacena los documentos fuente u originales de manera completa o contiene toda la información necesaria para dar respuesta a las necesidades del usuario.

- **BD de texto completo**. Constituida por los propios documentos en formato digital. También puede incorporar campos con información complementaria para facilitar la descripción y el acceso. Permite localizar términos presentes en el texto del documento. Se puede considerar un caso específico de BD multimedia donde sólo se almacenan y manipulan fuentes de texto.

- **Archivo electrónico de imágenes**. Constituido por referencias que tienen un enlace a la imagen del documento original. No permite localizar términos presentes en el texto original.

Ejemplos de sistemas de BD documentales

Algunos de los sistemas de BD documentales más usados son Knosys, Verity y Excalibur.

A modo de resumen, a continuación os presentamos una tabla con todos los tipos de bases de datos:

Para saber más

Esta tipología de BD se desarrolla en el apartado "Bases de datos documentales", al final de este módulo. También se estudia en otras asignaturas de la licenciatura.

Tipo de BD	Acrónimo	Estructura de datos	Estándar del modelo	
Jerárquicas	BDJ	**Árbol**		IMS, System 2000, SABRE
En red	BDX	**Red**	Codasyl-DBTG	IDS
Relacionales	BDR	**Tabla**	SQL	DB2, dBASE, Ingres, Oracle, Informix, Paradox, Access, Sybase, MySQL
De objetos	BDO/ BDOO	**Objeto**	ODMG	ObjectStore, Ardent (O2), Objectivity, Versant, Ontos
Objeto-relacionales	BDOR	Tabla, objeto		UniSQL/X, Illustra, OpenODB, PostgreSQL, Ominiscience
Activas	BDA	Tabla con *triggers*		Oracle, DB2 y Sybase (con *triggers*)
Deductivas	BDD*	Tabla con reglas lógicas		LDL, Validity
Temporales	BDT	Tabla u objeto con series de tiempo y versiones de *tuples* o de atributos	TSQL	Informix Universal Server con *datablades* de series de tiempo
Espaciales	BDE			
Información geográfica	SIG			ARC/Info, ARC/Storm
Multidimensionales	BDMD	Hipercubo		UniVerse (de IBM)
Paralelas	BDP			
Distribuidas	BDD*	Heterogénea	ODBC, JDBC	
Web	BD-Web	Heterogénea	ODBC, JDBC	
XML	BDXML	Documentos XML (semiestructurados)		Tamino, X-Hive/DB, dbXML, eXist, Ozone XML DB
Multimedia	BDMM	Objetos LOB, TDA		Sybase, Oracle, DB2, ODB II, CA-Jasmine, DS Informix, QBIC
Documentales	BDD*	Documentos extensos (textuales o no)		Knosys, Verity, Excalibur

El acrónimo BDD se utiliza para una u otra tipología según el entorno del que se trate. Los dos primeros tipos son obsoletos.

2.5. Bases de datos relacionales

Para saber más

Para tener más información sobre los diferentes tipos de bases de datos, podéis consultar el apartado "Tipos de bases de datos" de este mismo módulo.

Tal y como se ha comentado en los apartados anteriores, las bases de datos más extendidas actualmente son las relacionales, sobre todo con respecto al uso de particulares y departamentos de organizaciones.

Algunas entidades bancarias y hospitalarias todavía trabajan, sin embargo, en el ámbito corporativo con bases de datos prerrelacionales.

En este apartado se presentan los conceptos principales de una base de datos relacional y sus componentes: tablas, registros, campos, etc.

En una base de datos relacional, la información se almacena en tablas, compuestas por registros. Cada registro contiene distintos valores, almacenados en campos, que son de diferentes tipos de datos. Todos los registros se identifican con una clave. Las diferentes relaciones entre los registros de dos o más tablas se implementan por medio de claves foráneas.

2.5.1. Tablas

Una tabla es un conjunto de datos que agrupan todas las ocurrencias o elementos de un mismo tipo.

Para cada entidad de información diferente se utiliza una tabla distinta. Por ejemplo, todos los alumnos de la universidad, las asignaturas de la carrera o la lista de profesores.

2.5.2. Registros

Cada fila de la tabla es un registro de información y describe un elemento de este conjunto de informaciones.

Ejemplos de registros

Ejemplos de registros serían cada alumno de la universidad, cada una de las asignaturas o cada uno de los profesores, almacenados en las tablas de alumnos, asignaturas y profesores.

2.5.3. Campos

Cada valor o información de los registros posible es un campo.

Ejemplos de campos

Por ejemplo, en las tablas que hemos mencionado antes de alumnos y profesores de la universidad, el nombre, el apellido, el NIF, la dirección, el código postal, la población o el teléfono serían campos.

O, por ejemplo, en la tabla de asignaturas, el nombre de la asignatura, su código o los créditos (carga lectiva) serían campos.

2.5.4. Claves

La clave de una tabla son uno o más campos que identifican, de manera única, los registros de la tabla.

A continuación, se describen los diferentes **tipos de claves** que puede tener una tabla:

- **Claves candidatas** (*candidat*e key): son todas las posibles claves que permiten identificar, de manera única, a cada uno de los registros de una tabla.

- **Clave primaria** (*primary key*) o principal: es la clave escogida, de entre las claves candidatas de una tabla, para identificar de manera única sus registros. Se acostumbra a escoger la clave candidata formada por el menor número de campos, y se suele poner al principio de la tabla. También recibe el nombre de identificador.

- **Claves alternativas** (*alternative* key) o secundarias: son las claves candidatas que no son escogidas como clave primaria de una tabla.

- **Clave foránea** (*foreig*n key), ajena o externa: es el campo (o conjunto de campos) que puede formar parte o no de la clave primaria de una tabla (denominada tabla hija o referenciante) que, a la vez, es clave primaria o clave alternativa en otra tabla (llamada tabla maestra o referenciada) con la que se relaciona. Permite establecer relaciones en cascada entre tablas y puede no existir en una tabla. Su objetivo principal es mantener la integridad de la información y es parte importante en la normalización de la BD.

Además, una **clave simple** está formada por un único campo; una **clave compuesta**, por más de un campo.

Aunque los registros de una tabla se pueden identificar por más de una posible combinación de sus campos, es decir, pueden existir varias claves candidatas, sólo una de éstas es escogida como clave primaria. El resto quedan como claves alternativas.

Una tabla puede tener N claves candidatas, 1 clave primaria, N-1 claves alternativas y M claves foráneas, donde N es igual o mayor que 1 y M igual o mayor que 0. Dicho de otra manera, una tabla tiene, como mínimo, una clave candidata (que, en el peor de los casos, estará formada por todos los campos de la tabla y será escogida como clave primaria). La clave primaria, normalmente, es un pequeño subconjunto de los campos de la tabla. Además, una tabla puede tener ninguna o varias claves foráneas procedentes de diferentes tablas referenciadas.

Para saber más

El ejemplo de la relación uno a muchos, expuesto en el siguiente subapartado, "Relaciones", muestra la clave foránea CódigoDepartamento de la tabla Profesor necesaria para implementar la relación con la clave primaria de la tabla Departamento.

Ejemplos de claves

Siguiendo con los ejemplos anteriores, los alumnos se pueden identificar, de manera única, por su NIF o por el código interno en la universidad. Ambos campos serían claves candidatas. Una de ellas sería escogida como clave primaria y la otra quedaría como clave alternativa. Además, el nombre y apellidos no pueden ser clave, ya que no se puede garantizar que no haya dos alumnos que se llamen igual.

La clave primaria de una tabla permite relacionar los registros de ésta con los de otras tablas por medio de las claves foráneas, tal como se explica a continuación.

2.5.5. Relaciones y claves foráneas

Las relaciones entre los registros de las diferentes tablas se implementan mediante la vinculación de valores. Esta vinculación se establece entre la clave foránea de una tabla y la clave primaria de otra.

El **tipo de relación** entre dos tablas se indica mediante el número de registros de una tabla, que se pueden relacionar con un registro de la otra tabla y viceversa. Eso se puede expresar separando por el signo de dos puntos (":") estos valores. Es decir, *1:1*, *1: n* o *m: n*, según el caso, donde las variables *n* y *m* simbolizan *muchos*.

Los tipos de relación entre dos tablas entroncan con el concepto de cardinalidad de una relación entre entidades (modelo conceptual), que es perfectamente aplicable a las relaciones entre tablas en el modelo relacional.

Para saber más

En el subapartado "El modelo entidad-relación", del apartado "Diseño de bases de datos", se desarrolla y se dan ejemplos de este concepto aplicado a las relaciones entre entidades.

Entre dos tablas se pueden establecer tres **tipos de relación**:

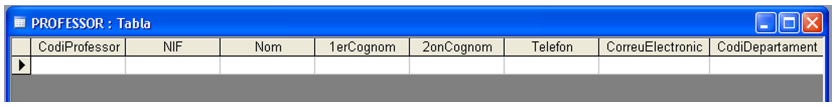
a) Uno a uno. A cada registro de una tabla le corresponde un solo registro de la otra. Se representa por *1:1*.

b) Uno a muchos. A cada registro de la primera tabla (la tabla maestra) le corresponde uno o más registros de la segunda (la tabla hija). Se representa por *1: n o n: 1* (la simbología utilizada por Microsoft Access es *1: ¥ y ¥ :1*).

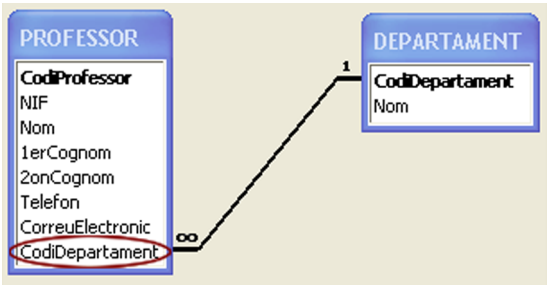
Ejemplo de relación uno a muchos (y de clave foránea)

Para implementar la relación que hay entre los profesores de un centro de estudios y el departamento al que están adscritos, se define, en la tabla Profesor, una clave foránea que se relaciona con la clave primaria de la tabla Departamento.

De esta manera, en el registro de cada profesor habrá un campo donde se almacenará el código del departamento en el que está adscrito, tal como muestra la tabla Profesor creada con el SGBD Microsoft Access de la siguiente figura.



Eso permite relacionar las tablas Profesor y Departamento, tal como se ve a continuación.

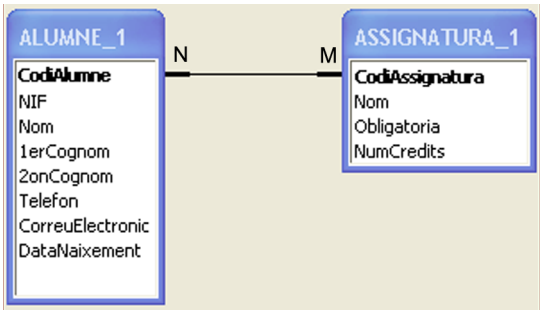


La figura muestra las claves primarias en negrita, y la relación uno a muchos (*1: ¥*) entre las dos tablas mediante una línea que une la clave primaria de la tabla maestra Departamento (en este extremo, se pone el número 1) y la clave foránea de la tabla hija Profesor (en este extremo, se pone el signo ¥).

c) Muchos en muchos. A cada registro de la primera tabla le pueden corresponder varios registros de la segunda y viceversa. Se representa por *m: n*.

Ejemplo de relación muchos a muchos

La relación que hay entre los alumnos de un centro de estudios y las asignaturas en las que se matriculan (inscriben) es del tipo muchos a muchos, ya que un alumno se puede inscribir en distintas asignaturas y, en cada asignatura, se pueden inscribir diferentes alumnos.



La figura muestra este tipo de relación *m:n*, no permitida en un SGBD relacional.

Los SGBD relacionales no permiten definir este tipo de relación entre dos tablas. Si se presenta el caso, hay que crear una tercera tabla, denominada **tabla intermedia**, que se relacione con las otras dos mediante relaciones *1: n* y *n: 1*.

Ejemplo de generación de una tabla intermedia

Para implementar la relación muchos a muchos del ejemplo anterior, hay que elaborar la tabla intermedia "Matriculación", que tendrá como mínimo dos campos: el identificador

de la asignatura y el identificador del alumno, siendo el conjunto de los dos campos su clave primaria.

Si, por ejemplo, un alumno se matricula más de una vez en una misma asignatura, ya no hay bastante con esta clave primaria. Habrá que añadir un tercer campo que ayude a identificar las distintas matriculaciones. Este tercer campo puede ser, por ejemplo, la fecha de matriculación. Ahora, la clave primaria estará formada por tres campos.

La figura creada con Microsoft Access muestra las relaciones *1: n* y *n: 1* existentes entre las tres tablas. La tabla intermedia aparece rodeada, y los campos que forman la clave primaria, en negrita.

2.5.6. Tipos de datos

Los campos de una tabla separan los tipos de información que contiene.

La persona que crea la tabla (diseñador de la base de datos) define los campos que tendrá esta tabla, es decir, qué informaciones se almacenarán para cada registro.

Los diferentes sistemas de gestión de bases de datos (SGBD) ofrecen una variedad de diferentes tipos de datos.

A continuación, definimos los principales.

- **Texto**: cadena o serie de caracteres alfanuméricos no muy larga. Puede ser información textual (el nombre de una persona o de una empresa, una dirección) que incluya letras, dígitos decimales, signos de puntuación, espacios y/o otros símbolos imprimibles. También puede ser un número (teléfono, código postal, etc.) que será tratado como una serie de dígitos y no como el valor numérico que representa y que, por lo tanto, no se podrá utilizar para hacer cálculos.

- **Numérico**: número que requiere cálculos matemáticos (exceptuando si está relacionado con un valor monetario; en tal caso, se utiliza el tipo "moneda").

- **Moneda**: valor numérico con un formato de presentación configurado para representar valores monetarios con decimales. Se utiliza para evitar el redondeo de decimales en los cálculos (A MS Access, tiene una precisión de hasta 4 decimales).

Formato de visualización

Es la forma en que se presentan (por pantalla o impresos) los valores de los diferentes tipos de datos. Los SGBD suelen proporcionar formatos de visualización predefinidos para los tipos: numérico, fecha/hora, moneda y sí/no. Además, se pueden definir formatos personalizados para todos los tipos de datos, excepto para el tipo binario.

- **Fecha y hora**: información temporal que representa fechas y horas.

- **Lógico**: cualquier tipo de dato en el que sólo haya dos valores posibles (sí o no, verdadero o falso, activado o desactivado, encendido o apagado, aprobado o suspenso, etc.). Se almacena uno de los dos valores posibles como 1 o 0, respectivamente.

- **Memo**: información textual alfanumérica más o menos extensa de longitud variable y de tipo memorando. Por ejemplo, descripciones, comentarios, notas, resúmenes, etc.

- **Binario**: objeto o archivo binario como una imagen, audio, vídeo, documento de texto con formato (creado con un procesador de texto), hoja de trabajo (creada con un gestor de hojas de cálculo) o cualquier otro tipo de archivo binario creado en otros programas que se vincula o se incrusta como dato en el campo de la BD mediante el protocolo OLE (*object linking and embedding*) de vinculación e incrustación de objetos.

- **Autonumérico**: valor numérico entero único que se genera, automáticamente, al agregar cada nuevo registro. Puede ser secuencial (con incrementos de una unidad; el primer registro tiene el valor 1, el segundo el valor 2, y así sucesivamente) o aleatorio. Una vez generada, no se puede actualizar y, si se elimina, el valor no se puede volver a usar. El tipo incremental, también denominado contador, es el más común y adecuado para utilizar como identificador único o clave primaria.

- **Hipervínculo**: direcciones URL o rutas de acceso según la convención universal de asignación de nombres, UNC).

El **tipo de dato** para utilizar en un campo de una tabla (valores del campo) se puede **decidir en función de**:

- El tipo de valores que se quieren permitir en el campo. Por ejemplo, no se puede almacenar texto en un campo definido para tipos de datos numérico.

- El tipo de operaciones que se quiere realizar con los valores del campo. Por ejemplo, se pueden sumar los valores de campos de tipo numérico y moneda, pero no los de campos de tipo texto o binario.

- La posibilidad de ordenar los valores del campo. Por ejemplo, los valores de un campo de tipo binario no se pueden ordenar.

- El sistema de ordenación de los valores del campo. Por ejemplo, en un campo de tipo texto, los números se ordenan como cadenas de caracteres (por ejemplo, 1, 10, 100, 2, 20, 200), no como valores numéricos (1, 2, 10, 20, 100, 200). Por un lado, para ordenar números como valores numéricos

conviene utilizar el tipo numérico o moneda. Por el otro, muchos formatos de fecha tampoco se ordenan correctamente si se utiliza un campo de tipo texto. Para garantizar un orden correcto, hay que usar un campo de tipo fecha/hora.

- El espacio de almacenaje que ocupan los valores del campo. Según las características de los datos textuales, se pueden usar tres tipos de datos de diferente tamaño:
  - Texto para almacenar texto corto: datos como nombres, direcciones y cualquier número que no necesite cálculos, como números de teléfono, números de producto o códigos postales. (En Access, hasta 255 caracteres).
  - Memo para almacenar texto extenso. (En Access, hasta 64.000 caracteres).
  - Binario para almacenar documentos de texto con formato más o menos largos.

En los campos con datos numéricos, se puede especificar la medida del campo en un rango de valores. Por ejemplo, en Access, para números enteros (sin decimales): byte: de 0 a 255, entero: entre -32.768 y 32.767, entero largo (predeterminado): entre -2.147.483.648 y 2.147.483.647.

2.5.7. Consultas

En este apartado, se comenta cómo los SGBD tratan la información almacenada en una BD, qué tipo de herramientas ofrecen para llevar a cabo esta tarea y las funcionalidades de las que disponen para consultar los datos.

Las consultas tienen funciones para seleccionar registros de las tablas y campos, o para hacer acciones de cálculo. Una prestación especial es la sentencia JOIN (de SQL), que permite combinar los registros de diferentes tablas, es decir, hacer consultas en las que intervienen varias tablas a la vez.

La sentencia JOIN

Es una instrucción de SQL que permite combinar (concatenar) registros (filas) de dos o más tablas en una BD relacional, si cumplen una condición de emparejamiento. Es un componente de la instrucción Select. Matemáticamente, se trata de una función de composición relacional, la operación fundamental en el álgebra relacional.

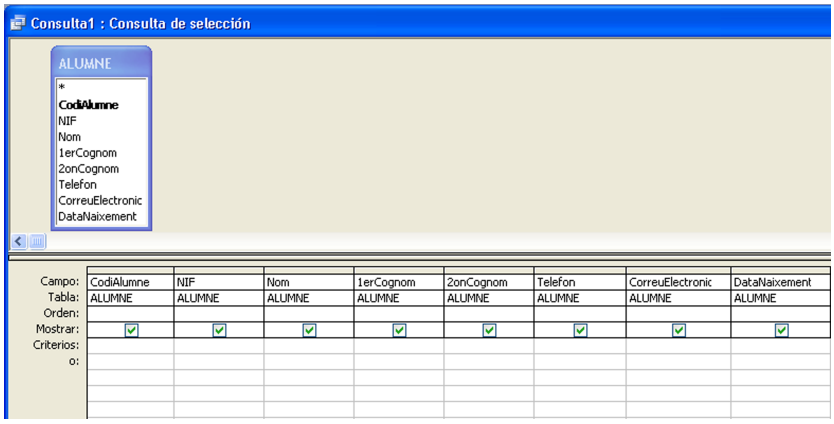
Ejemplos de diseño de consultas

A pesar de no ser objeto de esta asignatura, a continuación se ejemplifica el diseño de consultas (con Microsoft Access).

Se consultan los datos de los alumnos de un centro de estudios y las asignaturas en las que se matriculan.

1) Diseño de una consulta para obtener todos los datos de los alumnos del centro

En el diseño de una consulta en Access, hay que distinguir dos áreas diferenciadas. En el panel superior, se sitúan las tablas (con los campos) que intervienen en la consulta y se muestran las relaciones que hay entre ellas. En el panel inferior, se detallan los campos específicos que tienen que aparecer en la consulta y la tabla a la que pertenecen; además, otros aspectos (cómo se ordenarán los datos, si se tiene que mostrar o no la columna, los criterios de filtrado, etc.).



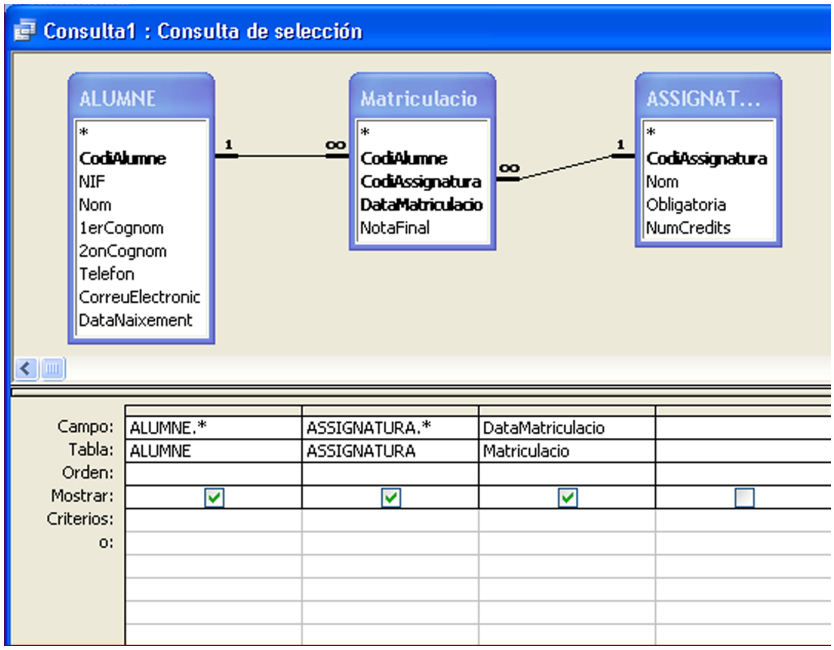
Para saber más

En el subapartado "Tablas intermedias", podéis ver el ejemplo de referencia de la BD de un centro de estudios donde se muestra cómo se relacionan las tablas Alumno, Asignatura y Matriculación.

2) Diseño de una consulta para obtener todos los datos de las matriculaciones del centro

En la fila de campos, en el panel inferior de esta consulta, aparecen los nombres de dos tablas acompañados de un asterisco. Eso quiere decir que se mostrarán todos sus campos.

Por lo tanto, el resultado de la consulta mostrará todos los datos de los alumnos y de las asignaturas a las que se han inscrito, además de la fecha de matriculación.



2.6. Lenguajes de bases de datos relacionales

Cualquier SGBD dispone de un lenguaje de BD que posee dos módulos para realizar funciones diferentes:

- El **lenguaje de definición de datos** (DDL, *data definition language*) especializado en la creación de la BD; es decir, en la definición del esquema de la BD.

- El **lenguaje de manipulación de datos** (DML, *data management language*) especializado en la utilización y la gestión de la BD; es decir, en el mantenimiento de la información y la realización de consultas.

Para comunicarse con el SGBD, el usuario, ya sea un programa de aplicación o un usuario final, se vale igual de este lenguaje de BD. Hay muchos lenguajes diferentes según el **tipo de usuario** para el que están pensados y lo que éstos tienen que poder expresar:

- Los **usuarios informáticos** muy expertos (programadores) que quieren escribir procesos complejos necesitarán lenguajes complejos.

- Los **usuarios finales esporádicos** u ocasionales que sólo hacen consultas necesitarán un lenguaje muy sencillo, aunque dé un rendimiento bajo en tiempo de respuesta.

- Los **usuarios finales cotidianos**, especializados o incluso dedicados, exclusivamente, a trabajar con la BD necesitarán lenguajes muy eficientes y compactos especializados en tipos concretos de tareas, aunque no sean fáciles de aprender.

El lenguaje de consulta estructurado **SQL** (*structured query language*) se considera el lenguaje estándar de BD relacionales. Por esto, se estudiará el DDL y el DML que utiliza el SQL.

Las características principales de SQL son la sencillez, el carácter estándar y ser un lenguaje declarativo o no procedimental; o sea, para que SQL realice una acción concreta no se especifica cómo lo tiene que hacer, sino qué es lo que quiere obtener.

El lenguaje SQL tiene **instrucciones** de tres tipos:

- Instrucciones de **definición de datos** (tipo DDL). Por ejemplo, Create Table para definir las tablas, sus campos y las restricciones.

- Instrucciones de **gestión de datos** (tipo DML). Por ejemplo, SELECT para hacer consultas e, INSERT, UPDATE y DELETE para el mantenimiento de los datos (insertar, actualizar y borrar registros, respectivamente).

- Instrucciones de **control del entorno**. Por ejemplo, COMMIT y ROLLBACK para delimitar transacciones.

A continuación, se exponen las instrucciones más importantes de definición y de manipulación que utiliza el SQL.

2.6.1. Lenguaje de definición de datos (DDL)

El lenguaje de definición de datos proporciona órdenes para definir, eliminar y modificar tablas, y también para crear índices y vistas.

Las órdenes o instrucciones SQL más importantes del DDL son:

- **CREATE TABLE**: crea una tabla con los campos y sus tipos de datos. Su sintaxis es la siguiente:
  CREATE TABLE n_tabla (n_camp1 t_camp1 ... n_campN t_campN)
  Donde n_tabla es el nombre de la tabla que se creará, n_camp1 es el nombre del primer campo, t_camp1 es el tipo de datos del camp1.
  Cada campo tiene asociado un tipo de datos, siendo los más comunes: INTEGER para números enteros; DECIMAL (x,y) para números reales, donde *x* representa la longitud total del campo e *y* el número de decimales; CHAR (n) para cadenas de caracteres de longitud *n* (entre 1 y 255); DATE para fechas, etc.

- **ALTER TABLE**: añade nuevos campos a una tabla existente, los modifica o los borra. No es una orden estándar. Su sintaxis es la siguiente:
  ALTER TABLE n_tabla ADD (n_camp t_camp) para añadir campos a la tabla.
  ALTER TABLE n_tabla MODIFY (n_camp t_camp) para modificar campos de la tabla.
  ALTER TABLE n_tabla DROP n_camp para borrar campos de la tabla.

- **DROP TABLE**: borra el esquema de la BD, es decir, destruye tanto los datos contenidos en la tabla como la estructura de la misma. Su sintaxis es la siguiente:
  DROP TABLE n_tabla.

2.6.2. Lenguaje de manipulación de datos (DML)

El lenguaje de manipulación de datos está basado en el álgebra relacional e incluye órdenes para insertar, suprimir y modificar registros (filas) de la BD.

Las órdenes SQL más importantes del DML son:

- **INSERT**: inserta nuevos registros (filas) en una tabla. Su sintaxis es la siguiente:

INSERT INTO n_tabla VALUES (valor_camp1 ... valor_campN).

- **UPDATE**: modifica valores de determinados campos (columnas). Su sintaxis es la siguiente:
  UPDATE n_tabla SET camp=valor_camp.
  UPDATE n_tabla SET camp=valor_camp (WHERE condición)
  Si se especifica una condición, únicamente se actualizan los valores de los campos de los registros que cumplen la condición.

- **DELETE**: borra registros (filas) de la tabla especificada. Su sintaxis es la siguiente:
  DELETE FROM n_tabla
  Si no se especifica ninguna condición, se eliminarán todos los registros de la tabla. Es decir, deja la tabla vacía, como si se acabara de crear, pero no destruye la estructura. No se tiene que confundir con orden DROP que, además de borrar la información, también elimina la estructura.
  DELETE FROM n_tabla (WHERE condición)
  Si se especifica una condición, únicamente se borran los registros que satisfacen la mencionada condición. Esta es su forma más usual.

- **SELECT**: es el orden principal de SQL y se utiliza para consultar tablas. Es muy flexible y admite muchas variaciones. Selecciona un conjunto de registros (filas) de una o más tablas que cumplan una condición. Si no se especifica la condición, se seleccionan todos los registros de la tabla. También permite crear un filtro seleccionando sólo algunos campos de la tabla. Su sintaxis es la siguiente:
  SELECT FROM n_tabla (WHERE condición) (ORDER BY campX) para seleccionar todos los campos de la tabla. Esta es su forma más usual.
  SELECT camp1, campN FROM n_tabla (WHERE condición) (ORDER BY campX) para consultar únicamente un conjunto de campos.
  SELECT camp1, campN FROM n_tabla1, n_tablaM (WHERE condición) (ORDER BY campX) para consultar campos de más de una tabla.

2.6.3. Herramientas de interfaz

Aunque casi todos los SGBD del mercado tienen el **SQL como lenguaje nativo**, ofrecen otras posibilidades:

- **Lenguajes de cuarta generación** (4GL, de *4th generation languages*) de muy alto nivel, que suelen combinar elementos procedimentales con elementos declarativos. Pretenden hacer muy fácil no tan sólo el tratamiento de la BD, sino también la definición de menús, pantallas y diálogos.

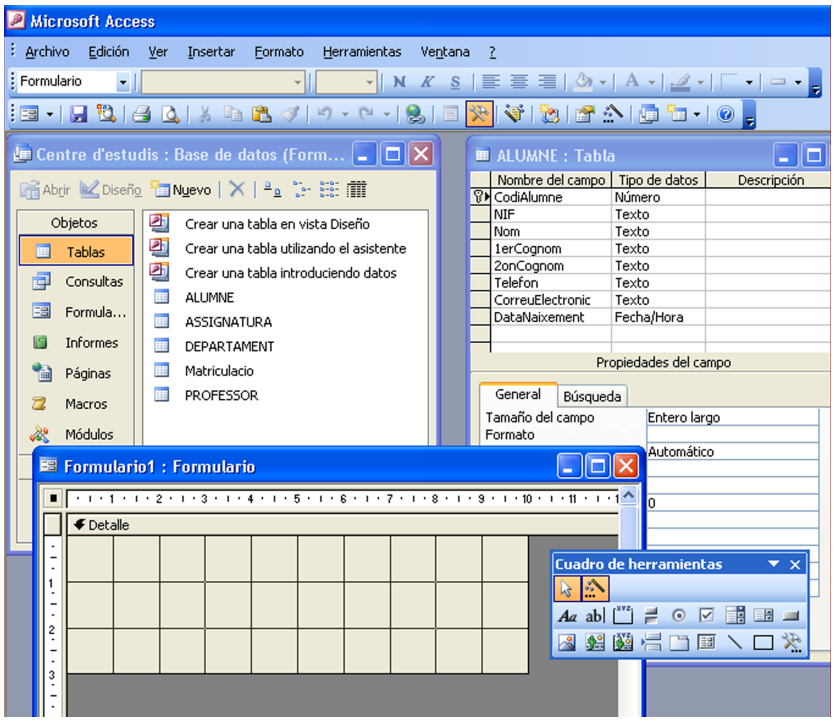
- **Interfaces visuales** muy fáciles de utilizar, que permiten utilizar las BD siguiendo el estilo de cuadros de diálogo con ventanas, iconos y ratón, típico de las aplicaciones en entorno Windows. No sólo son útiles a los

usuarios no informáticos, sino que facilitan mucho el trabajo a los usuarios informáticos: permiten consultar y actualizar la BD, y también definirla y actualizar la definición con mucha facilidad y claridad. Aparecieron a finales de los años ochenta y, desde entonces, han proliferado.

Ejemplo de interfaz visual

Un claro ejemplo de herramienta con interfaz visual es Microsoft Access, que permite crear los diferentes elementos de una BD (que aquí se denominan objetos: tablas, consultas, formularios, informes...) para gestionar y extraer la información mediante ventanas (*windows*) sin tener que utilizar, si no se quiere, el lenguaje SQL.

La siguiente imagen muestra algunos elementos de la interfaz de usuario de este SGBD (el panel de de exploración, una tabla y un formulario).



2.6.4. Programación de bases de datos

Si se quiere escribir un programa de aplicación que trabaje con la BD, seguramente se utilizará el mismo lenguaje de programación utilizado para desarrollar la aplicación informática (Pascal, Cobol, Basic, C, etc.). Sin embargo, generalmente, estos lenguajes no tienen instrucciones para las BD. Entonces, se puede recorrer en una de las siguientes **dos opciones**:

- **Llamamientos a funciones**: en el mercado, hay librerías de funciones especializadas en BD; por ejemplo, las librerías ODBC. Sólo hay que incluir llamamientos a las funciones deseadas dentro del programa escrito con el lenguaje de programación habitual. Las funciones se encargarán de enviar las instrucciones (generalmente, en SQL) en tiempo de ejecución en el SGBD.

- **Lenguaje hospedado**: otra posibilidad consiste en incluir directamente las instrucciones del lenguaje de BD en el programa. Pero eso exige utilizar

un precompilador especializado que acepte, en el lenguaje de programación habitual, las instrucciones del lenguaje de BD. Entonces, se dice que este lenguaje (casi siempre el SQL) es el lenguaje hospedado o incorporado (*embedded*), y el lenguaje de programación (Pascal, C, Cobol, etc.), el lenguaje anfitrión (*host*).

2.7. Resumen

Una **base de datos** es un conjunto de datos organizado que permite acceder, administrar y actualizar sus contenidos con facilidad.

Las BD evitan los datos redundantes, garantizan la independencia de los datos con respecto a los programas que las utilizan, almacenan los datos junto con sus relaciones y se puede acceder a ellos de diferentes maneras. Es decir, resuelven la mayoría de problemas que presentan los ficheros.

Las principales **características** que tiene que cumplir un sistema de BD son:

- Es accesible simultáneamente para diferentes usuarios, cada uno a una determinada información.
- Se controla el acceso de diferentes usuarios a los datos y se garantiza la confidencialidad y la seguridad.
- Los datos se pueden almacenar sin redundancia.
- Permite utilizar diferentes métodos de acceso y asegurar flexibilidad en las buscas.
- Dispone de mecanismos de recuperación de información en caso de que falle el hardware.
- El soporte físico donde se almacenan los datos se puede cambiar sin que se resientan éstos ni los programas que los utilizan.
- La modificación de su contenido y de las relaciones entre los datos no afecta a los programas que las utilizan.
- Dispone de una interfaz de usuario que permite utilizarla de manera cómoda y flexible.

El **SGBD** (sistema gestor de bases de datos) es el conjunto de software destinado a la creación, gestión, control y manipulación de la información almacenada en una BD.

Entre las principales **funciones de un SGBD**, destacan:

- Definición del esquema de la BD, por medio de un **lenguaje de definición de datos** (DDL).
- Acceso a la información desde un lenguaje de alto nivel denominado **lenguaje de manipulación de datos** (DML).
- Acceso a la información en modo conversacional por medio de una **interfaz de usuario**.
- Interacción con el sistema operativo para trabajar con los ficheros de datos que gestiona, mediante un módulo denominado **gestor de datos**.
- Los **objetivos de un SGBD** son los siguientes:
- Permitir realizar **consultas no predefinidas y complejas**.
- Dar **flexibilidad** a los cambios y obtener **independencia** entre los datos y los programas.
- Facilitar la **eliminación de la redundancia** para evitar inconsistencia e incoherencia de datos.
- Permitir el **acceso concurrente de los usuarios** a la misma BD.
- Permitir definir **autorizaciones** y **derechos de acceso** a diferentes niveles para garantizar la **confidencialidad** y la **seguridad** de los datos.

Históricamente, se han diferenciado tres tipos de BD:

- **BD jerárquicas**. Sistemas, aparecidos en los años sesenta, donde la información se representaba en forma de árbol. Su principal inconveniente era que no todas las BD se adaptaban a esta estructura.
- **BD en red**. Aparecieron para resolver el problema de las anteriores. Permitían cualquier tipo de relación y, por lo tanto, representar cualquier conjunto deinformación. Su principal inconveniente era su falta de flexibilidad.

- **BD relacionales**. Aparecieron, en los años setenta, para obtener una mayor flexibilidad en el tratamiento de los datos y son las más utilizadas desde los años ochenta.

Otros tipos de BD son los siguientes:

- **BD de objetos**. Almacenan la información en clases y subclases de objetos completos (estado y comportamiento) e incorporan conceptos de la programación orientada a objetos.

- **BD relacionales extendidas**. Son evoluciones del modelo relacional para obtener una mayor capacidad expresiva, incorporando funcionalidades de otros modelos de datos. Se distinguen:
  - **BD relacionales con frontal OO**: añaden una capa de OO sobre un SGBD relacional.

  - **BD objeto-relacionales**: integran los dos modelos, incorporando nuevos tipos de datos, operaciones para gestionar el comportamiento, nuevas capacidades consultiva y expresiva.

  - **BD activas**: permiten definir reglas que se activan cuando suceden determinados acontecimientos, que inician la ejecución de una acción si se dan unas condiciones.

  - **BD deductivas**: incluyen mecanismos de inferencia basados en reglas deductivas que permiten generar información adicional a partir de los hechos almacenados en la BD.

- **BD temporales**. Soportan la variable tiempo y otros conceptos temporales. Permiten almacenar un historial de cambios y consultar el estado (actual o pasado) de la BD.La información temporal se puede referir a hechos puntuales o a hechos de duración.

- **BD espaciales**. Proporcionan conceptos para interpretar y especificar las características espaciales de objetos que se encuentran en un espacio multidimensional. Pueden ser cartográficas (incluyen conceptos geométricos bidimensionales) y meteorológicas (incluyen información de puntos espaciales tridimensionales).

- **Sistemas de información geográfica**. Permiten almacenar, analizar y representar gráficamente información, que describe propiedades geográficas, gestionada por una BD que contiene datos espaciales (físicas, políticas, administrativas) y datos no espaciales (demográficos, económicos, comerciales).

- **BD multidimensionales**. Organizan los datos en estructuras matriciales de varias dimensiones (por ejemplo, productos, zonas comerciales, perio-

dos impositivos) y disponen de lenguajes especiales para realizar consultas complejas. Facilitan la visualización de datos, en cualquier combinación de dimensiones, mediante diferentes criterios, jerarquías (niveles de agrupamiento en categorías más generales o de disgregación en categorías más concretas) y operaciones de cálculo intensivo (agregación, suma, media).

- **BD paralelas**. Aprovechan el paralelismo de unidades de proceso para el procesamiento paralelo de gran número de tareas de consulta y actualización mediante la utilización concurrente de dos o más procesadores y/o dispositivos de almacenamiento. Eso mejora significativamente el tiempo de transacción y de respuesta en BD de grandes dimensiones.

- **BD distribuidas**. Reparten los datos entre diferentes ordenadores conectados en red y se basan en una arquitectura cliente-servidor. Tienen la ventaja que descentralizan la información y la hacen más disponible localmente, pero el inconveniente de que la gestión es más complicada.

- **BD accesibles por Internet**. Permiten el acceso flexible, homogéneo, eficiente y seguro, desde cualquier plataforma, en BD distribuidas y heterogéneas (que contienen tipo de datos muy distintos como, por ejemplo, datos multimedia) actualizadas permanentemente y almacenadas en servidores web, evitando problemas de compatibilidad de formatos y sistemas.

- **BD XML**. Almacenan documentos XML (que son estructuras semiestructuradas) con la eficiencia de las BD convencionales. Los SGBD XML pueden ser extensiones del modelo relacional o de objetos (que desglosan el documento XML) o nativos (que respetan la estructura del documento y lo recuperan tal como fue guardado originalmente).

- **BD multimedia.** Ofrece características que permiten almacenar y consultar información multimedia que incluye imágenes, vídeo, audio y documentos de texto. Su aplicación abarca disciplinas como la gestión documental, la difusión de conocimientos, la comunicación, el entretenimiento, el trabajo colaborativo y el control de actividades en tiempo real.

- **BD documentales o textuales**. Almacenan las referencias y/o el texto completo de documentos de propósito general que pueden contener texto extenso y datos multimedia. Se consultan por medio de palabras clave o de términos de un *thesaurus* de estructura jerarquizada.

Se pueden clasificar en BD referenciales (que contienen referencias para poder localizar los documentos originales), BD de texto completo (que almacenan el contenido completo de los documentos originales) o archivos electrónicos de imágenes (constituidos por referencias que tienen un enlace a la imagen del documento original).

Una **BD relacional** está formada por **tablas, que son estructuras en filas** (los registros de información) y **columnas** (los campos de información de los registros). Cada tabla tiene una **clave**, que es un campo o conjunto de campos (columnas) que identifica de manera única cada registro (fila), y permite relacionar la tabla con otras tablas de la BD.

Una **tabla** de una BD relacional cumple las siguientes **condiciones**:

- Todas sus filas son registros del mismo tipo. Para almacenar registros de diferentes tipos se utilizan diferentes tablas.
- Cada columna se identifica por un nombre de campo.
- No acepta nombres de campos (columnas) repetidos.
- No acepta registros (filas) duplicados.
- El orden de los registros es indiferente.
- Su contenido es independiente del soporte de almacenaje físico de los datos.
- Se relaciona con otras tablas haciendo coincidir valores iguales de los registros de ambas, mediante claves.

Ejemplos destacados de SGBD relacionales son: Microsoft Access, el servidor de BD Oracle, Microsoft SQL Server o MySQL.

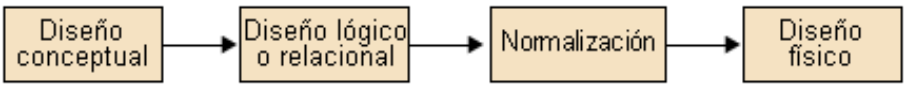
3. Diseño de bases de datos

Un error muy frecuente a la hora de diseñar una base de datos relacional es generar el modelo relacional (tablas, campos, etc.) directamente a partir de la identificación de las necesidades que tiene el usuario u organización, utilizando el sistema gestor de bases de datos.

Este método puede ser válido cuando se trata de crear bases de datos muy sencillas y con poca complejidad (pocas tablas y con pocas relaciones).

En cambio, cuando la complejidad aumenta, se debe recurrir a unas metodologías que garanticen la definición correcta de nuestro sistema de información y a unas estructuras que también faciliten que no haya errores y que simplifiquen el trabajo posterior, el tratamiento de los datos y la evolución del sistema.

La metodología o las etapas recomendadas son las siguientes:



a) Diseño conceptual

Durante el diseño conceptual, se construye un esquema de la información que se utiliza en la organización (empresa, asociación, departamento, etc.) independientemente de cualquier consideración física. Es decir, independientemente de qué modelo de bases de datos se utilizará posteriormente y en qué sistema gestor de base de datos se implementará.

El esquema resultante lo denominamos **esquema conceptual**.

En este paso, se describen las entidades, atributos y relaciones de la información que se debe almacenar posteriormente y que han de gestionar los sistemas informáticos.

La notación más popular para crear el esquema conceptual es el modelo entidad-relación, que se describe en el apartado siguiente.

Una vez se tiene el modelo conceptual, se puede generar el diseño lógico.

b) Diseño lógico

El diseño lógico es el proceso de construir un esquema de la información basándose en un modelo de base de datos (en red, jerárquico, relacional, etc.) e independientemente, sin embargo, del SGBD que después se utilizará para gestionar la información.

En el marco de este módulo, entenderemos por modelo lógico la creación del modelo relacional.

En esta etapa, se transformará el esquema conceptual en el conjunto de tablas, campos e interrelaciones entre las tablas con el objetivo de poder crear más tarde las bases de datos.

c) Normalización

Después de este paso, y antes de crear el esquema físico, es conveniente normalizar la base de datos, una técnica que se utiliza para verificar que las tablas obtenidas no tienen datos redundantes y están "optimizadas". En los próximos apartados se comentarán estos aspectos.

d) Diseño físico

El diseño físico es el proceso de generar la base de datos utilizando un sistema gestor de base de datos correspondiente.

En este paso, se llevan a cabo todas las actividades necesarias para crear las tablas, definir los campos, las interrelaciones entre tablas, índices, etc.

Creación de índices

Un índice permite ordenar y encontrar registros con mayor rapidez. Se puede indexar un único campo o una combinación de campos. La clave primaria de una tabla se indexa de forma automática. Determinados campos no se pueden indexar debido al tipo de datos que contienen.

En el marco de este material, sólo se tratará, a modo de introducción, el diseño conceptual y logicorrelacional y se introducirán conceptos sobre la normalización.

3.1. Modelos de datos: conceptos básicos

El objetivo básico de un sistema de BD es obtener, mediante la abstracción del mundo real, un conjunto estructurado de datos y un conjunto de operaciones definidas sobre ellas que permitan satisfacer, de forma eficiente, las necesidades de información de una organización.

La representación de una parcela de la realidad mediante un modelo de datos da lugar a un esquema. Sin embargo, el mundo real también comprende aspectos dinámicos, ya que los datos varían en el tiempo. Por eso, los modelos de datos se componen de dos submodelos que proporcionan elementos (o conceptos básicos) para representar la naturaleza estática y dinámica del sistema:

- **Naturaleza estática**: características inalterables que identifican al sistema y a sus objetos. Corresponden a lo que se entiende por estructura.

- **Naturaleza dinámica**: acciones que admite el sistema haciéndolo evolucionar y cambiar de estado a lo largo del tiempo. Describen el comportamiento de la estructura.

Un **modelo de datos** es el conjunto de conceptos, reglas y convenciones que permiten describir una parcela del mundo real que interviene en un problema determinado (de universo de discurso).

Los elementos que proporcionan los modelos de datos, a pesar de estar definidos con terminología y formalismo diferentes, tienen significados equivalentes. Eso permite agruparlos, genéricamente, en:

1) Elementos permitidos:

- Describen las **estructuras de datos** (los objetos, sus propiedades) y la forma en la que se relacionan (asociaciones).

- Definen las **operaciones** (y transacciones) que permiten la manipulación de los datos (adición, eliminación, modificación y recuperación).

2) Elementos no permitidos:

Son las **restricciones** o limitaciones impuestas a la estructura del modelo o a los datos, que invalidan determinadas ocurrencias en la BD. Se distinguen dos tipos:

a) Restricciones inherentes: son limitaciones de utilización en la misma naturaleza del modelo de datos, que no admite determinadas estructuras. Pueden ser de dos tipos:

- **Propias del modelo de datos**: su definición corresponde al diseñador, pero su gestión es responsabilidad del modelo de datos, que las reconoce y las recoge en el esquema.

- **Ajenas al modelo de datos**: el modelo de datos no las reconoce ni proporciona instrumentos para manipularlas; son responsabilidad del diseñador.

Ejemplos de restricciones inherentes

a) Propias del modelo de datos:

- El modelo relacional no permite atributos multivalor.
- Utilizar una regla de validación (restricción CHECK) para comprobar que la edad de los votantes es mayor de 18 años.

b) Ajenas al modelo de datos: las restricciones de cardinalidad mínima.

b) Restricciones semánticas o de integridad: son limitaciones impuestas a los valores de los atributos o a las características de las interrelaciones para reflejar, fielmente, la realidad. Permiten captar la semántica del universo de discurso que se quiere modelar, y verificar la corrección de los datos almacenados en la BD.

Ejemplos de restricciones semánticas

- El estado civil de una persona no puede pasar directamente de soltero/a a viudo/a.
- Una persona no puede tener una profesión si es menor de 18 años.
- El salario de un trabajador no puede ser mayor que el de su jefe o supervisor.

Las restricciones garantizan la integridad de la BD y la validez semántica de su contenido.

Todo **modelo de datos** comporta un modelado o proceso de abstracción que es la labor intelectual mediante la que se representa la realidad, con el fin de obtener una estructura para almacenar los datos.

Los **tipos de abstracción** básicos son los siguientes: clasificación/instanciación, generalización/especialización, agregación/desagregación y asociación/disociación.

3.2. Definición conceptual de una BD

Una base de datos es cara y difícil de crear.

La inclusión de una o más bases de datos en nuestras organizaciones implicará la necesidad de tiempo de preparación del software, la necesidad de contar con un experto para que diseñe la base de datos y un coste de mantenimiento.

Estas tareas se pueden llevar a cabo internamente o por medio de la contratación de terceros, externos a la organización.

Con el objetivo de generar una base de datos, se establece un proceso o metodología que se inicia con la visión del mundo exterior, en concreto, de la parte que nos interesa representar en datos.

En este proceso se debe aprender, comprender y conceptualizar este mundo exterior y transformarlo en un conjunto de ideas y definiciones que representen una imagen fiel del comportamiento del mundo real.

Este proceso de abstracción genera lo que conocemos como el **modelo conceptual**.

La definición correcta del modelo conceptual de la base de datos es imprescindible para garantizar la generación correcta de la posterior base de datos.

A continuación, se enumeran los componentes y elementos principales de una metodología para definir un modelo conceptual de datos: **entidades**, **atributos**, **claves** y **relaciones**.

3.2.1. Modelo de datos: el modelo entidad-relación

El modelo entidad-relación se basa en el uso de un conjunto de símbolos gráficos que permiten representar "la realidad" de la información que se quiere gestionar.

De esta manera, se identifican diferentes elementos de información como son las **entidades** y sus **atributos**, las **relaciones entre entidades** y la **cardinalidad** y el **grado** de estas relaciones.

El correcto diseño conceptual de la base de datos, siguiendo las normas y reglas de este modelo, facilitará posteriormente la definición y creación de la base de datos relacional.

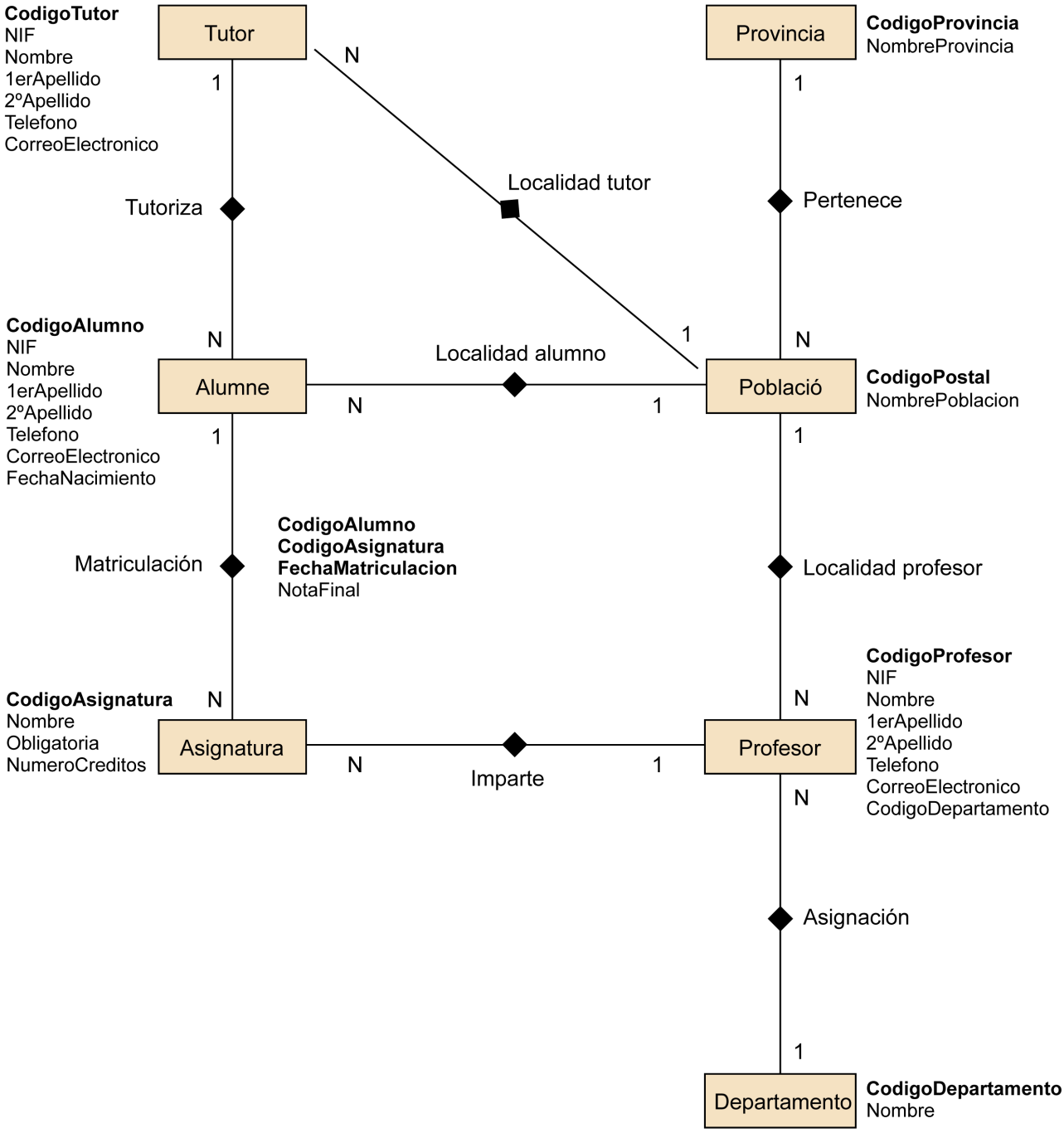
En este apartado, se definen los principales elementos de este modelo y en los casos prácticos del módulo se ejemplariza su uso.

Ejemplo de modelado

El siguiente es un posible modelado de la BD relacional de un centro de estudios (con departamentos, profesores, asignaturas, alumnos...) expuesta en los ejemplos de Bases de datos relacionales" del apartado anterior.

Para saber más

Otros ejemplos de modelación conceptual (generación del diagrama entidad-relación) se incluyen al final, en el subapartado "Casos prácticos de diseño y creación de bases de datos".



Para simplificar el modelado del ejemplo, se hacen los siguientes supuestos o restricciones semánticas:

- Una asignatura sólo puede ser impartida por un profesor.

- Los tutores no pueden ser profesores.

Equivalencia de los elementos del modelo relacional y del modelo entidad-relación

La tabla siguiente muestra la equivalencia entre los elementos básicos del modelo relacional y los del modelo entidad-relación.

	Modelo de BD		
	Relacional	**Entidad-relación**	Ejemplos
Elementos	Tabla	Entidad	Alumno, asignatura, profesor
	Registro	Ocurrencia	Joan Soler Pou, estadística
	Campo	Atributo	Nombre, apellidos, NIF, dirección
	Clave	Clave	NIF
	Relación	Relación	Matriculación (relación alumno-asignatura)

3.2.2. Entidades

Una entidad es un objeto real o abstracto con características diferenciadoras de otros objetos y cuya información se almacenará en la base de datos.

Para saber más

Los ejemplos presentados a continuación, para ilustrar los elementos del modelo entidad-relación, tienen un paralelismo directo con los expuestos para cada elemento del modelo relacional en el apartado "Bases de datos relacionales".

Toma como significado conceptos u objetos que tienen un papel importante en la compañía u organización.

Una entidad está formada por un conjunto de elementos de datos o atributos, cada uno de los cuales aporta alguna característica a la definición de la entidad.

Ejemplos de entidades

Estableciendo una analogía con los ejemplos vistos anteriormente, ejemplos de entidades serían Alumnos, Asignaturas y Profesores.

3.2.3. Ocurrencia

Por ocurrencia se entiende cada uno de los elementos que representa una entidad. Es decir, cada uno de los alumnos o profesores o cada una de las asignaturas sería una ocurrencia de las entidades Alumnos, Profesores y Asignaturas.

3.2.4. Atributo

Un atributo es una unidad básica e indivisible de información sobre una entidad que sirve para identificarla o describirla.

Ejemplo de atributos

Por ejemplo, el nombre, los apellidos, la dirección, el NIF, etc. serían atributos de la entidad Alumnos.

Normalmente, se utiliza la denominación **atributo** para especificar el nombre de atributo, que se tiene que diferenciar del valor de atributo o **valor.**

Ejemplos de valor

55.123.708-H, Joan, Soler Pou, 36 y Mayor, 51, 2.º-1.ª son, respectivamente, los valores de los atributos DNI, nombre, apellidos, edad y dirección.

Los atributos (y los campos) pueden ser de diferentes tipos:

- Atributo compuesto: dirección (se puede dividir en calle, número, piso y puerta) o fecha de nacimiento (día, mes y año).

- Atributo simple (o atómico): calle, año de nacimiento.
- Atributo derivado (o calculado): edad.
- Atributo almacenado: fecha de nacimiento.
- Atributo monoevaluado: edad.
- Atributo multievaluado: teléfono.

Un valor nulo puede tener varias interpretaciones:

- Valor no aplicable: el atributo no se aplica en esta ocurrencia de entidad (o registro). Por ejemplo, si una persona no tiene DNI.
- Valor desconocido: falta (por ejemplo, si no se conoce la estatura); no se sabe si existe (por ejemplo, correo electrónico); o se conoce, pero está ausente, todavía no se ha registrado.

3.2.5. Clave

Se denomina *clave de una entidad* al atributo o conjunto de atributos que permite identificar, de manera única, un elemento de una entidad.

Ejemplo de clave

El NIF podría ser, por ejemplo, la clave de las entidades Alumnos y Profesores.

3.2.6. Relaciones

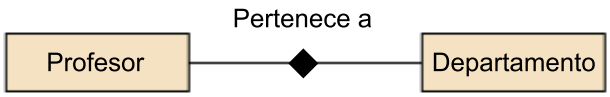
Las entidades, por sí solas, no describen la realidad de un sistema de información.

No es suficiente con identificar objetos; además, se han de definir las asociaciones que se dan entre los objetos o entidades.

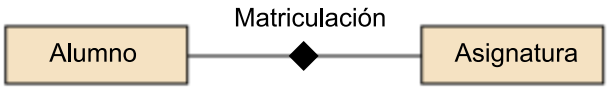
Estas asociaciones se denominan *relaciones*.

Ejemplos de relación

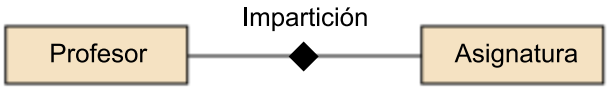
- La relación "Pertenece a" (o "Pertenencia a") que hay entre las entidades PROFESOR y DEPARTAMENTO.



- La relación "Matriculación" (o "Matriculado en") que se establece entre las entidades ALUMNO y ASIGNATURA.



- La relación "Impartición" existente entre las entidades PROFESOR y ASIGNATURA.



Toda relación presenta dos características que la definen: el grado (número de entidades que participan en la relación) y la cardinalidad de cada uno de ellas en la relación. Se describen a continuación.

Grado de una relación

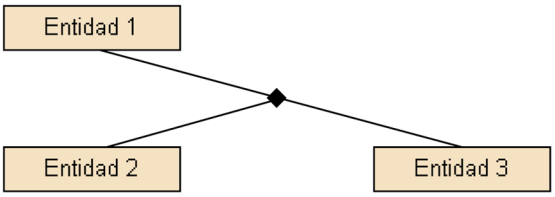
Por este concepto, se identifica el número de entidades que se interrelacionan.

La mayoría de las veces, el grado de una relación será binario (entre dos entidades, como las vistas anteriormente) o ternario (entre tres entidades, tal y como se verá en uno de los ejemplos posteriores del material).

Ejemplo de relación binaria y ternaria

Relación binaria

Relación ternaria



Se podría dar el caso de relaciones cuaternarias o superiores, pero son casos complejos y poco frecuentes.

Cardinalidad de una relación

La cardinalidad o el grado de una relación representa la participación en la relación de cada una de las entidades afectadas.

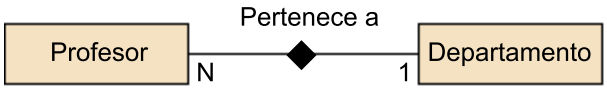
Eso, gráficamente, se indica poniendo sobre la línea que une las entidades el grado de participación (1, N o M) según corresponda a cada entidad.

En una relación binaria, hay tres tipos posibles:

- **Una a muchas** (*1:n*). A cada ocurrencia de la primera entidad le corresponde una o varias ocurrencias de la segunda.

Ejemplo de relación *1:n*

La relación "Pertenece a" tiene una cardinalidad "1 a n", ya que cada profesor pertenece a 1 único departamento y a un departamento pertenecen varios profesores (N).

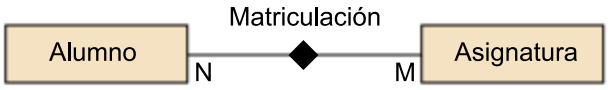


Eso se indica gráficamente poniendo, en la línea que une las entidades, una N junto a la primera entidad y un 1 junto a la segunda. En este caso, la primera entidad es DEPARTAMENTO (1) y la segunda PROFESOR (N).

- **Muchas a muchas** (*m:n*). A cada ocurrencia de la primera entidad le pueden corresponder diferentes ocurrencias de la segunda, y viceversa.

Ejemplo de relación *m:n*

La relación entre las entidades ALUMNO y ASIGNATURA, que se puede denominar Matriculación, tiene una cardinalidad "n a m", ya que cada alumno se puede matricular en más de una asignatura (M), y en cada asignatura se matricula más de un alumno (N).



- **Una a una** (*1:1*). A cada instancia u ocurrencia de una entidad le corresponde una y sólo una ocurrencia de la otra. Este caso se da pocas veces.

Ejemplo de relación *1:1*

Este caso se presenta al dividir en dos una entidad para separar/agrupar sus datos en dos tipos de entidad diferentes.

3.3. Del modelo conceptual al lógico o relacional

Una vez diseñado el modelo conceptual de una base de datos, se puede generar el modelo relacional a partir de este modelo.

A continuación, se incluye un resumen de las reglas de conversión principales:

- Toda entidad del modelo conceptual se convierte en una tabla en el modelo relacional. Por ejemplo, alumnos, profesores, asignaturas, etc.

- Todo atributo de una entidad se transforma en un campo de la tabla. Por ejemplo, de la entidad Alumnos se generan los campos Código, Nombre, Primer apellido, Segundo apellido, Teléfono, etc.

- Toda relación 1:*n* se convierte en una clave foránea en la tabla correspondiente a la entidad de grado *n*. Por ejemplo, la relación Adscripción entre profesores y departamentos.

- Toda relación *m*:*n* se transforma en una tabla intermedia. Por ejemplo, la tabla correspondiente a la relación Matriculación.

Para saber más

Podéis ver la conversión de la relación *1:n* ("adscripción", entre PROFESOR y DEPARTAMENTO) en clave foránea y de la relación *m:n* (matriculación, entre ALUMNO y ASIGNATURA) en tabla intermedia en los ejemplos expuestos en el ítem "Relaciones" del subapartado "Bases de datos relacionales".

Etapas de diseño conceptual y lógico de la base de datos

En esta asignatura sólo se abordarán, desde un nivel práctico, los pasos siguientes del diseño:

a) Diseño del modelo conceptual (modelo entidad-relación). Identificación de las entidades, sus atributos y las interrelaciones entre ellas, con los atributos de éstas, si fuera el caso.

b) Generación del modelo relacional (modelo lógico). Partiendo del modelo conceptual diseñado en el apartado anterior, éste se tiene que convertir en una BD relacional, indicando las tablas correspondientes, los campos de cada uno, el tipo de dato de cada campo, la clave primaria de cada tabla y las claves foráneas correspondientes.

Pautas para denominar entidades, atributos, tablas y campos

Los nombres de entidades y atributos (y, por extensión, de tablas y campos) pueden incluir cualquier combinación de letras, números, espacios en blanco y caracteres especiales excepto el punto (.), el signo de admiración (¡!), el acento grave (`) y los corchetes ([]), entre otros. Además, en Microsoft Access, las tablas y campos tampoco pueden empezar por un espacio en blanco ni incluir comillas dobles (") y, en total, pueden tener hasta 64 caracteres.

a) Consejos:

- A pesar de que se pueden usar espacios, es mejor evitarlos, ya que se pueden producir conflictos si se hace referencia a nombres con espacios en expresiones o código de Visual Basic para aplicaciones.

- Evitar el uso de nombres extremadamente largos porque es difícil recordarlos y referirse a ellos.

- Evitar que el nombre coincida con el nombre de una propiedad o elemento que utilice el SGBD (Microsoft Access), pues podría producir un comportamiento inesperado de la BD.

b) Convenios:

- Entidades (y tablas) en singular y mayúscula.
- Atributos (y campos) en singular, iniciales de palabra en mayúscula y resto en minúscula.
- Si tienen varias palabras, éstas se juntan sin dejar espacios. Otra opción, menos recomendable, es usar el guión bajo (_) para separarlas.

Para saber más

Al final del módulo, hay dos casos resueltos de diseño de BD donde se ejemplifican las etapas de diseño conceptual y lógico.

3.4. Normalización de bases de datos

La normalización de bases de datos es el proceso mediante el cual se transforman datos complejos en un conjunto de estructuras de datos más pequeños, que además de ser más simples y más estables, son más fáciles de mantener.

También se puede entender la normalización como una serie de reglas (formas normales) que sirven para ayudar a los diseñadores de bases de datos a desarrollar un modelo relacional que minimice los problemas de lógica.

Cada regla o forma normal está basada en la anterior.

La normalización se adoptó porque la antigua manera de poner todos los datos en un lugar, tanto en un archivo como en una tabla de la base de datos, era ineficiente y conducía a errores cuando se querían manipular los datos.

La normalización también hace las cosas más fáciles de entender entre los desarrolladores de la base de datos y los usuarios que la requieren y plantean los requisitos.

Otra ventaja de la normalización de una base de datos es que ayuda a optimizar espacio. Una base de datos normalizada ocupa menos espacio que una no normalizada, ya que evitamos repeticiones de valores innecesarios.

3.4.1. Grados de normalización

El proceso de normalización de las bases de datos tiene un conjunto de reglas para cada fase.

Básicamente, hay tres niveles de normalización:

- Primera forma normal (1FN).
- Segunda forma normal (2FN).
- Tercera forma normal (3FN).

Cada una de estas formas normales tiene sus reglas.

Cuando una base se conforma en un nivel, se considera normalizada según aquella forma normal.

Algunas veces, tener una base de datos normalizada al más alto nivel la puede hacer más compleja que si no lo está.

Normalización de una base de datos

Algunas veces, la normalización completa de una base de datos podría devenir en un gran número de tablas. Se podría dar el caso, entonces, de que para obtener informaciones relacionadas, hubiera que unir muchas de estas tablas, por medio de una consulta, moderando el proceso. En estos casos sería más eficiente disponer de tablas no normalizadas pero con la información ya relacionada.

Como introducción, en la tabla siguiente se introduce cada una de estas tres formas normales.

Regla (forma normal)	Descripción
Primera forma normal (1FN)	Consiste en la eliminación de todos los grupos repetidos
Segunda forma normal (2FN)	Asegura que todas las columnas (campos) que no son clave, dependan completamente de la clave primaria
Tercera forma normal (3FN)	Elimina las dependencias transitivas. Una dependencia transitiva es aquélla en la que las columnas (campos) que no son clave dependen de otras columnas (campos) que tampoco son clave.

3.4.2. Primera forma normal (1FN)

La regla de la primera forma normal establece que las columnas repetidas se deben eliminar y colocar en tablas separadas.

De esta manera se resuelven los problemas de cabeceras de columna múltiples. En lugar de tener una única tabla inmensa que tiene muchos aspectos, disponiendo de distintas tablas más pequeñas y gestionables, cada una con los datos de un aspecto concreto.

En este caso, la normalización ayuda a clarificar la base de datos y organizarla en partes más pequeñas y más fáciles de entender.

3.4.3. Segunda forma normal (2FN)

La regla de la segunda forma normal establece que todas las dependencias parciales se deben eliminar y separar de sus propias tablas. Una dependencia parcial es un término que describe los datos que no dependen de la clave primaria de la tabla para identificarlas.

3.4.4. Tercera forma normal (3FN)

Una tabla está en la tercera forma normal si todas las columnas que no son clave primaria son total y funcionalmente dependientes de la clave primaria y no hay dependencias transitivas. Una dependencia transitiva es aquélla en la que hay columnas que no son clave primaria y que dependen de otras columnas que tampoco son clave primaria.

Cuando las tablas están en la tercera forma normal se evitan errores al insertar o borrar registros.

Cada campo en una tabla está identificado de manera única por la clave primaria y no hay datos repetidos.

3.4.5. Ejemplos

El objetivo de este módulo es, simplemente, introducir el concepto de normalización, pero no se espera que los estudiantes dominen las técnicas de diseño de bases de datos y su normalización, por lo que, a continuación, se enumeran dos casos como ejemplo para hacer más fácil la comprensión de los conceptos.

Ejemplo de normalización en 1 FN

Los datos de los socios de un club de ocio y de las actividades que han llevado a cabo se podrían recoger en una tabla con la siguiente estructura:

Base de datos sin normalizar (una tabla)

NIF	Núm. socio	Nombre	Apellidos	...	Actividad	Fecha actividad	...
1					Globo	01-06-10	
1					Salida a Cardona	05-07-10	
2					Globo	01-06-10	
2					Salida a Cardona	05-07-10	
3					Salida a Cardona	05-07-10	
3					Baloncesto	30-06-10	
4					Baloncesto	30-06-10	
5					Baloncesto	30-06-10	
...							

En la tabla, se puede comprobar que los socios con NIF 1 y NIF 2 participaron en las actividades de globo y salida a Cardona, y que el socio con NIF 3 participó en las actividades salida a Cardona y baloncesto.

La tabla no está normalizada, dado que se presentan grupos de datos repetidos. Por una parte, en cada registro de actividad de un socio se repiten todos sus datos personales (núm. socio, nombre, apellidos, etc.) y, por la otra, para todos los socios que han parti-

cipado en una misma actividad se repiten los datos de la actividad (nombre y fecha de realización).

Base de datos normalizada (1FN) (dos tablas)

Socio				
NIF	**Núm. socio**	**Nombre**	**Apellidos**	**...**
1				
2				
3				
4				
5				
...				

Actividad		
Actividad	**Fecha actividad**	**...**
Globo	01-06-10	
Salida a Cardona	05-07-10	
Baloncesto	30-06-10	

Para obtener la BD normalizada en 1FN, se han colocado los grupos de columnas que repiten datos en tablas separadas. De esta manera, ya no se repiten los datos de los socios ni los datos de las actividades.

El siguiente paso en el proceso de normalización, dado que la relación entre las tablas resultantes es muchos a muchos (un socio puede participar en más de una actividad, y cada actividad puede tener varios socios participantes), sería crear una tabla intermedia.

Ejemplo de normalización en 2FN

Una situación muy habitual en BD, que contiene información de personas y/o de organizaciones, es la relación existente entre los códigos postales y la población.

Siguiendo con el ejemplo anterior, al definir los datos de la tabla SOCIO también se incluye el código postal y la población, tal como muestra la tabla.

Base de datos sin normalizar (una tabla)

NIF	Nombre	Apellidos	....	Código postal	Población	...
1				08700	Igualada	
2				08700	Igualada	
3				08202	Sabadell	
4				08206	Sabadell	
5				08225	Terrassa	
6				08760	Martorell	

NIF	Nombre	Apellidos	....	Código postal	Población	...
7				08700	Igualda (*)	
8				08202	Sabadell	
9				08760	Martorel (*)	
...						

Se puede comprobar que casi todos los campos dependen de la clave primaria (que puede ser el NIF). En cambio, el campo población depende del código postal. Eso significa que la BD no está normalizada según la segunda forma normal.

Para normalizarla, hay que crear una segunda tabla (POBLACIÓN) donde haya un registro para cada código postal y población, y en la tabla SOCIO dejar el campo código postal (será clave foránea) que se vinculará con la tabla POBLACIÓN.

En la tabla, se puede apreciar que se repiten valores (del código postal y la población) en los registros correspondientes al NIF 1 y NIF 2, o al NIF 3 y NIF 8.

Además, en una situación real, al no estar la BD normalizada según la 2FN, puede haber datos equivocados. En el ejemplo, eso se evidencia en los registros de los NIF 7 y NIF 9, en los que el nombre de la población es incorrecto ("Igualda" y "Martorel", respectivamente, que se han marcado con asterisco), aunque el código postal está bien. En tal caso, si se hace una consulta, por el campo población, de los socios que viven en Igualada no se encontrará el registro del NIF 7. Lo mismo pasará con Martorell y el registro del NIF 9.

Base de datos normalizada (2FN) (dos tablas)

Socio					
NIF	Nombre	Apellidos	....	Código postal	....
1				08700	
2				08700	
3				08202	
4				08206	
5				08225	
6				08760	
7				08700	
8				08202	
9				08760	
...					

Población	
Código postal	Población
08700	Igualada
08202	Sabadell

Población	
Código postal	**Población**
08206	Sabadell
08225	Terrassa
08760	Martorell
...	

Una vez normalizada la BD, queda resuelto el problema de los datos repetidos en la tabla SOCIO, y también posibles errores como los detectados en la situación anterior.

3.4.6. Otras formas normales

Después de la tercera forma normal, hay algunas más (forma normal Boyce-Codd, cuarta forma normal, quinta forma normal o forma normal de proyección-unión, etc.), pero si se garantiza tener las bases de datos en tercera forma normal, se resuelven la mayoría de los problemas y las dificultades.

3.5. Casos prácticos "Diseño y creación de bases de datos"

A continuación, se plantean y se resuelven dos casos prácticos que ayudan a ejemplificar los conceptos expuestos en el presente apartado de diseño de BD. Uno corresponde a un centro de formación, y el otro a una empresa de alquiler de coches. En el segundo, hay un ejemplo de relación ternaria.

El diseño de una base de datos no es único, ya que depende de los supuestos y/o restricciones que haya que tener en cuenta.

En una situación real, es habitual que el diseñador de la BD solicite diferentes requerimientos al cliente (responsable y/o usuarios de la BD) para ir aclarando posibles dudas que se presenten durante el proceso de diseño.

Por lo tanto, los ejercicios propuestos no tienen una única solución, ya que en función del diseño del modelo conceptual que se haga, la BD relacional obtenida será una u otra.

3.5.1. Caso práctico 1: Base de datos "Centro de formación"

FormProf, S. A., Formación Profesional en Informática, es un centro de formación de las tecnologías de información, de creación reciente.

Sus servicios se basan en la planificación y la ejecución de programas de formación en el área de informática, tanto en el ámbito del usuario (introducción a Internet, uso de MSFT Outlook, trabajo con procesadores de textos, creación

de animaciones con Flash, etc.), como en el de "programadores especialistas" (desarrollo de aplicaciones con Visual Studio .NET, instalación y configuración de redes con Unix, etc.).

La empresa se acaba de crear y a vosotros os han contratado como responsables del área de informática. No como profesores, sino como responsables de los servicios internos de TI.

El equipo humano del centro y su organización es la siguiente:

- Un director, responsable máximo del centro y de sus operaciones.
- Departamento Comercial y de Marketing. Dependiente del director del centro, habrá un equipo de fuerza de ventas, compuesto por dos comerciales asesores que cubrirán todo el territorio, junto con una persona responsable de las campañas de marketing y promoción.
- Un responsable del área de RRHH y Finanzas.
- Un auxiliar administrativo que dará apoyo a las tareas de RRHH y Finanzas y Contabilidad.
- Un responsable tanto de la recepción y atención al cliente, como de dar apoyo al director.
- Finalmente, hay un responsable de tecnologías, que sois vosotros.

Como equipo docente, tenemos la organización siguiente:

- Un jefe de estudios que también ejerce como profesor puntualmente.
- Dos profesores, pertenecientes a la plantilla del centro, responsables tanto de la dirección de programas de formación, como de impartir parte de sus módulos.
- Un conjunto de profesores, ajenos al centro, contratados ocasionalmente para impartir algunos de los módulos y cursos ofrecidos por FormProf.

A continuación, se especifican los requisitos de una base de datos para almacenar y gestionar toda la información de las acciones formativas que se llevan a cabo en este centro, y también el seguimiento de los clientes, los alumnos, las sesiones de formación, etc.

- Por una parte, se debe almacenar la información de todos los **cursos** que se planifican y ejecutan en nuestra compañía. Para cada curso, se ha de guardar la información de su identificador (será un código de cinco letras), el título del curso, la fecha de inicio y la fecha de finalización y un valor,

del tipo Sí/No, que indicará si el curso es de calendario (valor Sí) o a medida (valor No) para una compañía.

- Así, se debe guardar la información de las **compañías** clientes, indicando su CIF, razón social, dirección, código postal, población, teléfono, correo electrónico y nombre de la persona de contacto.

- Un curso lo puede organizar una compañía (en el caso de que sea a medida) o de calendario. En este último caso, no tiene ninguna compañía asociada. Una compañía nos puede contratar más de un curso.

- Para cada curso, también se almacenará la información de los **asistentes** o matriculados. Es necesario disponer de la información siguiente: NIF, nombre, apellidos, dirección, teléfono y correo electrónico. Se ha de tener en cuenta que un alumno puede participar en más de un curso.

- Por otra parte, también debemos poder guardar la información de las **sesiones** de un curso. Todo curso tendrá un mínimo de una sesión (en caso de que sea de un solo día). Para cada sesión debemos guardar la información siguiente: código de la sesión (identificador numérico), hora de inicio, hora de fin y fecha de realización.

- Para cada sesión se debe guardar la información del **profesor** responsable. De cada profesor guardaremos el NIF, el nombre y los apellidos. Sólo habrá un profesor por sesión, pero, evidentemente, un profesor participará en distintas sesiones de formación.

- Y, finalmente, es necesario guardar **el aula** donde se llevará a cabo la sesión (Aula 1, Aula 2 o "In Company").

El correcto diseño de la BD tiene que permitir realizar consultas para obtener, a partir de la información almacenada, los siguientes resultados:

- Determinar qué cursos son de calendario.

- Especificar la empresa para la que se ha organizado cada uno de los cursos hechos a medida.

- Obtener la lista de todos los alumnos asistentes a un curso.

- Conocer qué profesores impartirán alguna sesión de un curso.

- Saber qué sesiones tiene que impartir un profesor entre unas fechas concretas.

- Identificar las sesiones que se tienen que llevar a cabo o se han de realizar en una fecha concreta.

Se pide realizar las siguientes actividades:

1) Diseño del modelo conceptual de la BD. Identificación de las entidades, sus atributos y las interrelaciones entre ellas, con los atributos de éstas (si fuera el caso) en un esquema entidad-relación.

2) Generación del modelo relacional de la BD. Partiendo del esquema conceptual diseñado en el apartado anterior, convertirlo a una BD relacional, indicando las tablas correspondientes, los campos de cada una, el tipo de dato de cada campo, la clave primaria de cada tabla y las claves foráneas correspondientes.

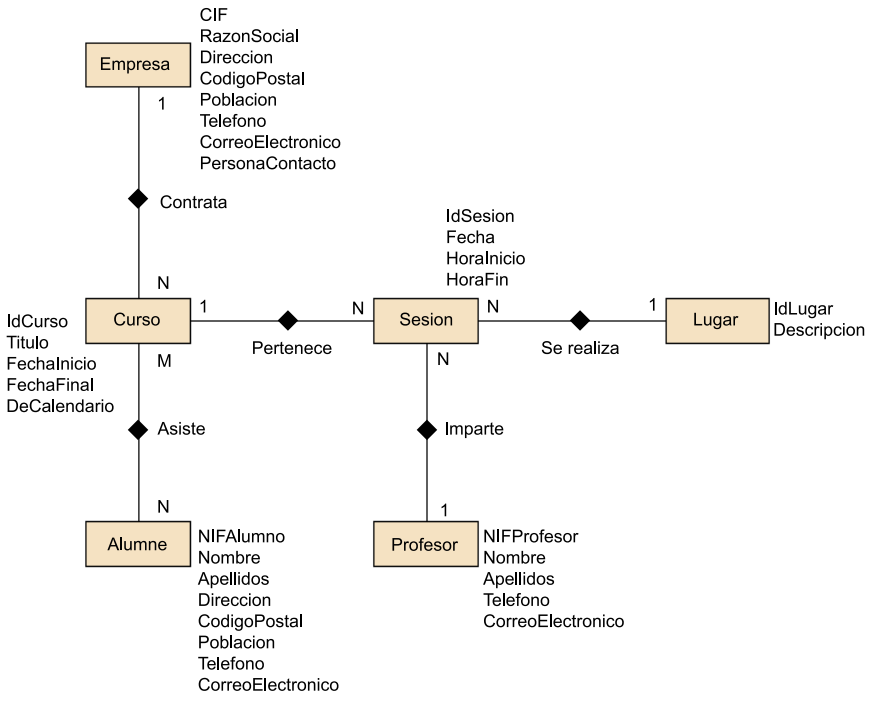
Caso práctico 1: Base de datos "Centro de formación". Resolución

Este ejercicio no tiene una solución única, ya que en función del diseño del modelo conceptual, la base de datos relacional final será una u otra.

En una situación real, durante el diseño de la base de datos, el responsable de ésta solicita diferentes requerimientos al "cliente" para ir determinando ambigüedades y dudas posibles.

1) Modelo conceptual de la base de datos

El modelo conceptual se representa, gráficamente, mediante el siguiente diagrama entidad-relación (DER), donde se identifican las entidades, sus atributos y las relaciones entre entidades:



Notas:

- Hay que destacar la relación *m:n* entre CURSO y ALUMNO, ya que normalmente un curso tiene diferentes alumnos y un alumno puede estar matriculado en varios cursos.

- En un modelo real, sería conveniente considerar la población como una entidad identificada por el código postal. De esta manera, no habría que guardar el nombre de la población en las entidades EMPRESA y ALUMNO, sino que la existencia de dos interrelaciones (empresa-población y alumno-población) permitiría guardar sólo el código postal. Este aspecto, sin embargo, no se ha considerado en el caso pronunciado con el fin de simplificar el diseño.

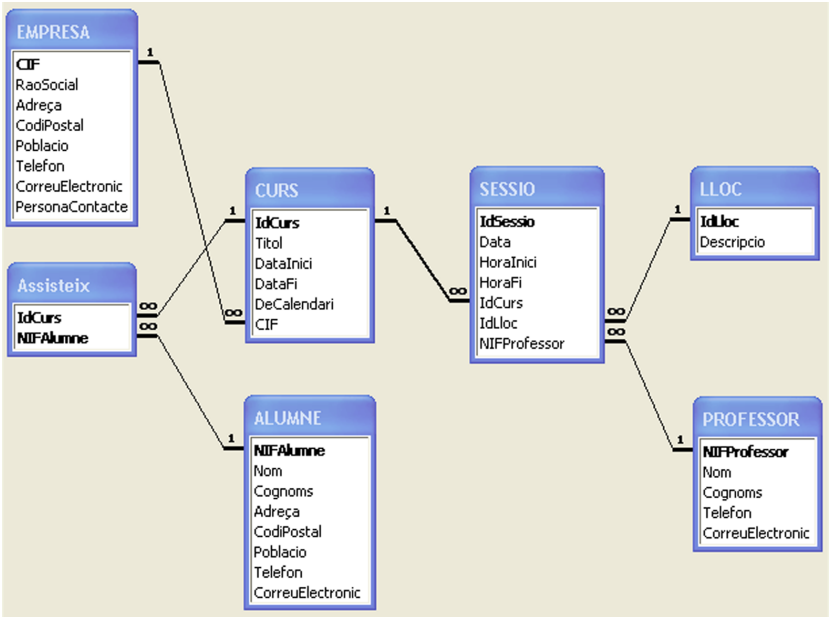
2) Generación modelo de base de datos relacional

Para generar este modelo, se han aplicado las siguientes reglas de conversión:

- La relación original *m:n* entre las entidades CURSO y ALUMNO se transforma en tres tablas. Una para CURSO, otra para ALUMNO y una intermedia que permite almacenar la información de la relación asiste (o alumno-curso) existente entre ellas.

- El resto de entidades se transforman en tablas; los atributos de cada entidad, en campos de cada tabla; y las relaciones *1:n* se establecen entre la clave primaria de una tabla (extremo 1) y la clave foránea de otra (extremo *n*).

La siguiente imagen muestra las tablas, sus campos y las relaciones entre las tablas del esquema relacional de la BD creado con MS Access. Los campos que forman clave primaria aparecen en negrita.



3.5.2. Caso práctico 2: Agencia de alquiler de coches

Se quiere diseñar una base de datos para almacenar y gestionar la información utilizada en una empresa dedicada al alquiler de vehículos, teniendo en cuenta los requisitos siguientes:

- La empresa dispone de un conjunto de vehículos para su alquiler. Se necesita conocer la matrícula, marca, modelo, color, tipo de combustible y tarifa diaria de alquiler de cada coche.

- Cada vehículo está asignado a un determinado garaje, que no puede cambiar, del que interesa conocer su código, la denominación, los datos de ubicación (dirección, población, código postal) y el teléfono.

- Los datos para almacenar de cada cliente son el NIF, el nombre, los datos de ubicación y los de contacto (teléfono y correo electrónico).

- Cuando un cliente desea alquilar un coche, se pone en contacto con alguna de las agencias o delegaciones de la empresa de alquiler para solicitar una reserva.

- De cada agencia se guardará su código, la denominación, los datos de ubicación y las de contacto.

- Un mismo cliente puede hacer o haber hecho varios reservas a lo largo del tiempo, que se pueden solapar en el tiempo.

- Un cliente puede hacer reservas en diferentes agencias. En cada reserva, participa una única agencia.

- Una reserva puede incluir más de un vehículo. De cada reserva se registrará la fecha de inicio (fecha de entrega del/de los vehículo/s al cliente), la fecha de finalización (fecha prevista para el retorno del/de los vehículo/s), el precio total de la reserva y un indicador de si el/los vehículo/s se ha/n devuelto.

- Para cada coche, interesa recoger el nivel de combustible que contiene el depósito en el momento de su entrega al cliente.

- El precio final de la reserva de cada coche se obtiene multiplicando la tarifa diaria de alquiler por el número de días que el cliente lo quiere reservar.

- El precio total de una reserva se obtiene sumando los precios finales de alquiler de todos los coches incluidos en la reserva.

El correcto diseño de la base de datos deberá permitir que la empresa pueda realizar consultas del tipo siguiente:

- Lista de todas las agencias, coches y clientes.

- Historial de todas las reservas realizadas indicando el cliente, las fechas y la agencia que lo ha tramitado.

- Número de reservas realizadas por los clientes de una población concreta.

- Resumen de las reservas (número, fecha inicio, fecha final y precio total) pendientes de devolución, ordenadas por fecha de finalización.

- Lista de los coches guardados en cada garaje.

- Agencias (código y nombre) que participan en reservas todavía no devueltas.

- Lista de todas las reservas que ha tenido un coche determinado.

- Cliente con más de tres reservas, alguna de las cuales la ha realizado la agencia con código "4".

- Para cada cliente de la compañía, lista de todas las reservas realizadas, así como el precio medio de todas ellas.

- Las agencias que no participan en ninguna reserva con fecha de inicio posterior a una fecha concreta.

- Todos los coches (matrícula, marca y modelo) reservados más de tres veces.

- La marca y el modelo del coche más reservado.

- Coches que no se han reservado nunca.

- Agencias que participan en más de tres reservas de una semana o más de duración.

- Precio de la reserva más cara realizada para cada marca y modelo de coche.

Caso práctico 2: Agencia de alquiler de coches. Resolución

En este apartado se presenta una solución posible para este enunciado, dando por hecho que puede haber otras.

1) Diseño del modelo conceptual

A continuación se indican las entidades y los atributos identificados en el modelo conceptual (se ha subrayado el atributo que identifica de manera única las instancias de las entidades):

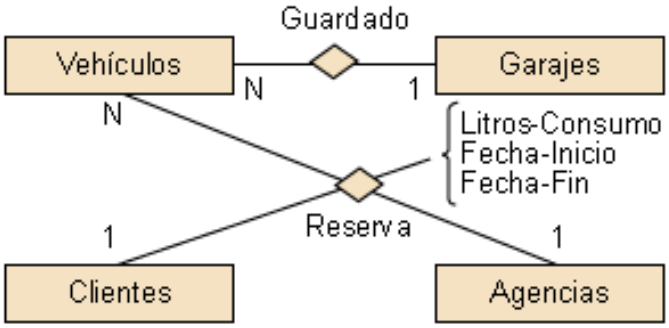
- Vehículos: matrícula, marca, modelo, color, tipo combustible y tarifa diaria.
- Garajes: código, denominación, dirección, código postal, población, teléfono.
- Clientes: NIF, nombre, apellidos, dirección, código postal, población, teléfono, correo electrónico.
- Agencias: código, nombre, dirección, código postal, población, teléfono, correo electrónico.

Por otra parte, se han identificado las relaciones siguientes entre las entidades:

- Guardado: Vehículos-Garajes
- Reserva: Vehículos-Cliente-Agencia

En el caso de la relación reserva, también se debe guardar una serie de atributos: litros-consumo, fecha-inicio, fecha-fin.

De manera gráfica, este modelo se representará del modo siguiente:



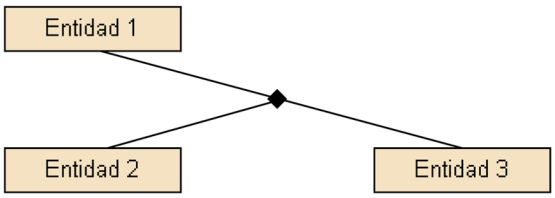
Modelo relacional

Al crear el modelo relacional, debemos tener en cuenta las consideraciones siguientes:

- Cada entidad será una tabla.
- El atributo de cada entidad es un campo de la tabla.
- El atributo que identifica de manera única las instancias de la entidad es la clave primaria de la tabla.
- La relación binaria Guardado, entre vehículos y garajes (1:n) se implementa por medio de una clave foránea en la tabla vehículos que "apunta" al código de garaje donde se encuentra aquel coche.
- La relación "ternaria" Reservas entre vehículos, cliente y agencia, se convierte en una tabla intermedia que contendrá las claves primarias de estas tres tablas, así como los litros consumidos por el vehículo concreto, la fecha de inicio de la reserva y la fecha de fin.
- La clave primaria de esta tabla será la combinación de las tres claves (vehículo, cliente y agencia) junto con la fecha de inicio de la reserva.

A continuación se presentan las diferentes tablas (creadas utilizando el MS Access):

Y en la imagen siguiente, se identifican las relaciones entre las diferentes tablas de la base de datos en MSFT Access:



Notas:

- También se habría podido realizar el diseño de la base de datos considerando "Reservas" como una entidad. Se ha preferido hacerlo de la otra manera para introducir un caso con una relación ternaria.
- Si se quisiera normalizar la base de datos deberíamos crear una tabla de Poblaciones, donde estaría almacenado el código postal y la población a la que corresponde. En las tablas Clientes, Agencias y Garajes no estaría el campo correspondiente a la po-

blación, sino el campo correspondiente al Código postal que "apuntaría" (sería clave foránea) a la tabla de Poblaciones.

Para saber más

Para tener más información sobre la normalización de las bases de datos, podéis consultar el apartado "Normalización de base de datos" de este mismo módulo.

3.6. Resumen

Un **modelo de datos** es el conjunto de conceptos, reglas y convenciones que permiten describir una parcela del mundo real que interviene en un problema determinado (de universo de discurso).

El diseño y la creación de una base de datos se caracterizan por lo siguiente:

- El diseño de una BD consiste en definir el modelo lógico; habitualmente, el **modelo relacional**. Es decir, construir un esquema de la información (crear las tablas, sus campos y las relaciones entre tablas), con independencia del SGBD que se utilizará.

- Previo al diseño de este modelo relacional, se aconseja hacer el **modelo conceptual**, que permite plasmar, en un modelo entidad-relación, las diferentes informaciones que se quieren almacenar (entidades, atributos e interrelaciones) con independencia de consideraciones físicas.

- Antes de crear el esquema físico, es conveniente aplicar una técnica de **normalización** de las tablas para verificar que éstas no tienen datos redundantes y están "optimizadas".

- Finalmente, en el **diseño físico** se implementan todos los elementos de la BD (tablas, campos, interrelaciones, índices, etc.) mediante un SGBD concreto.

Esquema de la conversión de elementos del modelo conceptual al modelo relacional

Modelo conceptual		**Modelo relacional**
Entidad	**se convierte en**	Tabla
Atributo	Campo	
Relación *1:n*	Clave foránea en la tabla correspondiente a la entidad de grado *n*	
Relación *m:n*	Tabla intermedia	

4. Bases de datos documentales

Tal como ya hemos visto en los apartados anteriores, una base de datos se compone de diferentes registros con su correspondiente número de identificación.

> Las bases de datos documentales **(BDD)** se caracterizan por que cada registro se corresponde con un documento, de cualquier tipo, publicación, documento gráfico o sonoro, etc., o con su referencia.

La información contenida en una base de datos documental se estructura en diferentes campos con el fin de facilitar el control a ella y el acceso individualizado.

Algunos campos se referirán a la descripción formal del documento y, en otros, se tratará el contenido temático. Incluso, en algunos casos, podrán registrar el mismo documento.

Para facilitar la rapidez en la recuperación de la información, las BDD facilitan la elaboración de diccionarios o índices alfabéticos. Algunos de estos sistemas trabajan con un índice único formado por palabras procedentes de diferentes campos de cada registro, mientras que en otros sistemas se gestionan índices para cada campo.

Las BDD se pueden clasificar en diferentes categorías según la información que contienen y la referencia al documento correspondiente: BD de texto completo (que contienen los propios documentos), archivos electrónicos de imágenes (que contienen imágenes de los documentos originales) y BD referenciales (que contienen referencias para localizar los documentos originales).

También se pueden clasificar por otras tipologías: según el **modo de acceso a la información**, la **cobertura documental** o según el **modelo de tratamiento documental**.

Por lo tanto, para utilizar eficazmente una BDD se recomienda adaptarse tanto como se pueda a sus características particulares.

Se debe estar informado del contenido y de cómo se han de efectuar las buscas, si cubre una o varias bases de datos, cuál es la cobertura temática, si es un catálogo o dispone de índices y resúmenes, etc.

4.1. Búsqueda de información

Todos los sistemas de recuperación de información permiten realizar diferentes modalidades de busca:

- **Directa**. Se teclean directamente una o varias palabras y se puede interrogar en texto libre o sobre campos individuales.

- **Por medio de índices**. En estos casos, el usuario visualiza un diccionario o índice alfabético de las entradas de todos los campos y selecciona los más adecuados. Se puede buscar por índices de palabras o por frase.

- **Jerarquizada**. La consulta se realiza mediante una estructura jerárquica. A partir de un concepto jerárquico, se pueden localizar no sólo los registros en los que aparece aquel término, sino todos aquéllos en los que figure algún concepto más específico de su campo semántico.

- **A través de código**. Numéricos o alfanuméricos que codifican la clasificación, el idioma o la tipología documental, entre otros.

Así pues, para realizar una busca hay que utilizar un número elevado de conceptos, para lo que hay distintas **herramientas** que construyen una estrategia y relacionan, de manera clara, los diferentes términos utilizados en la busca:

- **Operadores lógicos o booleanos**. Permiten la combinación de conceptos en una misma busca, tanto si es la unión (OR) como el resto (AND NOT), la intersección (AND) o la operación contraria (XOR).

- **Operadores sintácticos de proximidad**. Presencia en la misma frase, el mismo párrafo, aparición en un orden determinado o con una separación mínima, etc. Por ejemplo, NEAR.

- **Refinamiento**. Incluso, en algunos casos, el sistema permite refinar las buscas, aplicando una nueva al conjunto de registros encontrados en la busca anterior.

4.2. Categorías de BD según la información que contienen

Los registros de estas bases de datos pueden incluir o no el contenido completo de los documentos que describen, según el cual se describen tres categorías:

- **Bases de datos de texto completo**. Constituidas por los mismos documentos en formato electrónico (digital). Pueden incorporar, además, campos con información complementaria para facilitar su descripción y acceso. Permiten localizar términos presentes en el texto del documento.

- **Archivos electrónicos de imágenes**. Constituidos por referencias que permiten un enlace directo con la imagen del documento original. En estos casos, la busca está limitada a los campos de la referencia bibliográfica, pero no se pueden localizar términos presentes en el texto del documento original.

- **Bases de datos referenciales**. Son aquéllas en las que los registros no contienen el texto original sino sólo la información fundamental con el fin de describir y permitir la localización de documentos imprimidos, sonoros, iconográficos, etc. En estos casos sólo se pueden obtener referencias sobre documentos, que debemos localizar después en otro servicio (archivo, biblioteca, etc.). También pueden incluir campos que faciliten la localización del documento y/o enlaces directos para obtener directamente el original por medio de otro programa.

4.3. Tipología de BD según el modo de acceso

En función del tipo de acceso permitido a la base de datos, también se pueden clasificar en tres categorías: de acceso local, en dispositivo físico (CD-ROM, DVD, etc.) o en línea.

- **Bases de datos de acceso local**. Con el fin de consultarlas, hay que ir al centro productor, tanto a su biblioteca como al centro de documentación donde están almacenadas.

- **Bases de datos en dispositivos de almacenamiento**. Se adquieren, habitualmente, por compra o suscripción y están almacenadas, físicamente, en dispositivos como los CD-ROM o DVD.

- **Bases de datos "en línea"**. Se pueden consultar desde cualquier ordenador, ya que son accesibles "por Internet".

Para saber más

En el ítem sobre BD distribuidas del subapartado "Tipo de bases de datos", podéis ver las ventajas del acceso a BD en línea por medio de Internet y una descripción de la tecnología que se utiliza.

Una misma base de datos puede tener diferentes tipos de acceso.

Sin embargo, es posible que la actualización de ésta sea diferente en cada caso. Por ejemplo, una base de datos en CD-ROM se actualizará cada x meses o años, para volver a ser distribuida, mientras que una base de datos "en línea" puede estar actualizada en tiempo real.

4.4. Tipología de BD según la cobertura documental

Otra clasificación de las bases de datos documentales es la que se establece en función de los tipos de documentos que almacenan, centradas tanto en un único tipo de documento como en varios tipos.

- **Centradas en un único tipo de documento**. Recopilan y permiten la localización de un tipo de documento en concreto como, por ejemplo, patentes, tesis doctorales, informes, artículos de revista, etc.

- **En varios tipos de documentos**. Su objetivo es dar información sobre una disciplina e incorpora, por lo tanto, diferentes tipologías documentales. Se centran en una temática específica.

4.5. Tipología de BD según el modelo de tratamiento documental

Finalmente, una última clasificación de las BDD se realiza según el modelo de tratamiento documental, es decir, según las "operaciones" sobre la información del documento que el productor ha llevado a cabo.

- **Bases de datos de sumarios** o sin análisis de contenido. Se componen de referencias bibliográficas sencillas en las que el productor se limita a grabar los datos de la misma fuente y no realiza ningún análisis del documento. Sólo incorporan los datos descriptivos con el fin de localizar el documento, como, por ejemplo, el autor, el título y los datos de la fuente. En estas bases de datos, la busca por materias sólo se puede llevar a cabo por medio de las palabras del título del documento (normalmente artículos).

- **Catálogos de bibliotecas**. Estas bases de datos corresponden a fondos contenidos en una biblioteca o en una red de bibliotecas. Tienen una gran homogeneidad.

- **BD con análisis documental completo**. Sistemas de información que incorporan un número mayor de puntos de acceso con el fin de facilitar la localización por materias. Cada registro bibliográfico incluye un resumen del contenido del documento original y/o un conjunto de conceptos o términos representativos de los temas tratados. En este grupo de bases de datos se puede distinguir entre las que contienen clasificación y resúmenes, las que tienen una clasificación e indexación por descriptores o palabras clave, y las que disponen de una clasificación, una indexación y resúmenes.

- **Índices de citas**. Sistemas de información en los que además de extraer datos de descripción de los documentos, también se tratan las referencias bibliográficas citadas en los artículos de las revistas científicas.

Descriptor

Cada uno de los términos (o expresiones) escogidos entre un conjunto para representar o etiquetar (en calidad de término preferido) un concepto susceptible de aparecer con cierta frecuencia en los documentos indexables, y de ser utilizado en las consultas. Es la unidad mínima de significación que integra un *thesaurus*; el término por el que serán indexados y recuperados los documentos referidos a su temática.

Tesaurus (del latín *thesaurus*)

Lista alfabética de términos (palabras y expresiones) utilizados para representar los conceptos o temas de los contenidos de los documentos, con objeto de efectuar una normalización terminológica que permita un mejor acceso a los mismos.

Es un intermediario entre el lenguaje natural de los documentos y el lenguaje controlado usado por los especialistas de un determinado campo de conocimiento.

Los términos que conforman un *thesaurus* se interrelacionan entre sí bajo tres modalidades de relaciones:

- Jerárquicas: establecen subdivisiones que reflejan estructuras del tipo todo/parte.
- De equivalencia: controlan la sinonimia, homonimia, antonimia y polinimia entre los términos.
- Asociativas: mejoran las estrategias de recuperación y ayudan a reducir la multiplicidad de jerarquías entre términos.

4.6. Resumen

Las **bases de datos documentales** (BDD) se caracterizan por que cada registro se corresponde con un documento de cualquier tipo: publicación, documento gráfico o sonoro o a su referencia.

Almacena las referencias y/o el texto completo de documentos de propósito general que pueden contener texto extenso y datos multimedia que se consultan por medio de palabras clave (que aparecen en el texto) o de un *thesaurus* de estructura jerarquizada.

La información contenida en una BDD se estructura en diferentes campos para facilitar su control y acceso individualizado. Algunos campos se refieren a la descripción formal del documento y otros tratan sobre su contenido temático; incluso, pueden guardar el propio documento.

Se pueden clasificar en diferentes categorías:

- **Según la información que contienen**: BD referenciales (contienen información descriptiva y referencias que permiten localizar los documentos originales), BD de texto completo (guardan el contenido completo de los documentos originales) o archivos electrónicos de imágenes (contienen referencias que tienen un enlace en la imagen del documento original).
- **Según el modo de acceso a la información**: BD de acceso local, BD en dispositivos físicos de almacenaje y BD en línea.
- **Según la cobertura documental**: BD centradas en un único tipo de documento y BDcon diferentes tipos de documentos (centrado en una disciplina).

- **Según el modelo de tratamiento documental**: BD de sumarios (o sin análisis de contenido), BD con análisis documental completo, catálogos de bibliotecas e índices de citas.

Ejercicios de autoevaluación

Seleccionad para cada pregunta la respuesta adecuada.

1. Si existen las memorias (RAM y ROM), ¿por qué son necesarios los dispositivos para almacenar datos?

a) Porque la memoria RAM es volátil.
b) Porque la memoria ROM es permanente, no se puede cambiar (al menos no se puede cambiar de una manera sencilla y rápida).
c) Porque la RAM es muy cara y de capacidad limitada. Debemos contar con dispositivos con gran capacidad de almacenar datos, aunque paguemos el precio de una velocidad menor.
d) Todas son ciertas.

2. Señalad la única de las cuatro afirmaciones que es cierta:

a) La memoria ROM sólo se borra cuando se desenchufa el ordenador de la corriente eléctrica.
b) En la memoria RAM se puede leer y escribir.
c) Los dispositivos de almacenamiento de datos no son necesarios si el ordenador cuenta con memoria RAM.
d) Todas las afirmaciones anteriores son falsas.

3. Indicad cuál de las afirmaciones siguientes es verdadera:

a) La memoria ROM es sólo de lectura y de escritura bipolar.
b) La información almacenada en la memoria ROM se pierde si después de apagar el ordenador se desenchufa de la corriente eléctrica.
c) En la memoria RAM se puede leer y escribir información y perdura incluso si apagamos el ordenador.
d) Ninguna de las anteriores es cierta.

4. Indicad cuál de las afirmaciones siguientes sobre los dispositivos de almacenamiento de datos es cierta:

a) Permiten un volumen mayor de almacenaje de datos y con más rapidez que el ordenador.
b) Los diferentes dispositivos de almacenamiento existentes ofrecen diferentes prestaciones y capacidades de almacenaje.
c) Pierden su información cuando se desenchufan de la corriente eléctrica.
d) Permiten únicamente la escritura de datos, ya que la lectura se debe efectuar por medio de la CPU.

5. Indicad cuáles de las afirmaciones siguientes son falsas.

a) En los soportes secuenciales para acceder al registro **n** se deben leer necesariamente los **n – 1** anteriores.
b) A los soportes de acceso directo no se puede acceder directamente a menos que se conozca la dirección física.
c) Dentro de un fichero los registros se identifican por un único campo que es la clave.
d) La memoria intermedia es un espacio de la memoria principal que se utiliza en las operaciones de lectura y escritura en los ficheros.

6. Indicad cuál de las afirmaciones siguientes es cierta:

a) Los ficheros permanentes son los que, independientemente del tipo de soporte en el que estén, no desaparecen al apagar el ordenador.
b) Los ficheros permanentes son los que no sufren modificaciones en su contenido a lo largo del tiempo.
c) Los ficheros temporales sólo son necesarios para operar con la memoria principal.
d) Los ficheros de maniobras permiten ejecutar aplicaciones en las que los requerimientos de datos superan la capacidad de la memoria principal.

7. Indicad cuál es la única afirmación verdadera entre las cuatro siguientes:

a) En la operación de fusión de ficheros se unen dos o más ficheros para integrarlos en uno solo.
b) La operación de busca sobre un fichero consiste en llevar a cabo los pasos necesarios para encontrar el archivo dentro del disco duro del PC.
c) La eliminación de un registro es una operación de destrucción de ficheros.

d) Todas las afirmaciones anteriores son falsas.

8. Seleccionad la única afirmación verdadera entre las cuatro propuestas siguientes:

a) Hay dos tipos de acceso a los ficheros: secuencial directo y directo automático.
b) Exclusivamente, hay dos tipos de organización de ficheros: relativo y secuencial.
c) Hay cuatro tipos de organización de ficheros: secuencial, secuencial indexada, secuencial encadenada y rotativa.
d) Todas las afirmaciones anteriores son falsas.

9. Indicad cuál de las afirmaciones siguientes sobre ficheros es verdadera:

a) Un fichero se compone de campos que a su vez están compuestos de registros.
b) La información contenida en un fichero está desestructurada (de aquí la necesidad de las bases de datos).
c) El registro físico se corresponde con la cantidad de información que se puede leer y escribir al mismo tiempo en soporte físico.
d) Todas las anteriores son ciertas.

10. Indicad qué afirmación de las cuatro siguientes es verdadera:

a) Las bases de datos son una colección de información de tipo exclusivamente informático.
b) Las bases de datos evitan sin excepciones las redundancias de información.
c) Un sistema gestor de bases de datos (SGBD) accede a la información desde un lenguaje de alto nivel denominado "Lenguaje de manipulación de datos" (DML).
d) Todas las afirmaciones anteriores son falsas.

11. Indicad qué afirmación de las cuatro siguientes es verdadera:

a) Las bases de datos aseguran que no habrá redundancia en los datos que almacenan en ningún caso.
b) En algunos casos las bases de datos permiten redundancias con el fin de mejorar el espacio que necesitan para almacenarse.
c) Algunas veces las bases de datos presentan redundancias con el fin de mejorar los accesos y el rendimiento.
d) Todas las afirmaciones anteriores son falsas.

12. Seleccionad la afirmación falsa entre las cuatro siguientes:

a) Una base de datos es información almacenada de manera organizada en un ordenador.
b) La información contenida en una base de datos puede ser de cualquier tipo.
c) La utilización de una base de datos asegura la eliminación de las inconsistencias de datos.
d) La utilización de una base de datos presenta más ventajas que la utilización de un sistema de ficheros.

13. Indicad cuál de las afirmaciones siguientes es verdadera:

a) Los datos almacenados en una base de datos no presentan redundancias.
b) Un buen sistema de base de datos debe permitir realizar cambios sobre el soporte físico de la base de datos sin que por ello se vean afectados los programas que la utilizan.
c) Las bases de datos textuales son más modernas y más utilizadas que las jerárquicas y que las relacionales.
d) Todas las anteriores son falsas.

14. Indicad cuál de las afirmaciones siguientes es cierta:

a) Las relaciones entre diferentes tablas de una base de datos relacional se implementan mediante las claves internacionales.
b) Una tabla almacena toda la información de cualquier tipo que ha de contener la base de datos.
c) Para cada tabla debe haber una clave primaria que es uno y sólo uno de sus campos que no repite un valor en la tabla.
d) Todas las anteriores son falsas.

15. ¿Cuál de las afirmaciones siguientes es cierta?

a) Cuando la relación entre dos tablas es de uno a muchos es necesario crear una tercera tabla con el fin de poder representar el modelo físico.
b) En el modelo conceptual una entidad está formada por una serie de atributos.
c) En la generación del modelo físico a partir del conceptual toda relación 1:n se convierte en una clave foránea en la tabla correspondiente de grado 1.

d) Todas las anteriores son falsas.

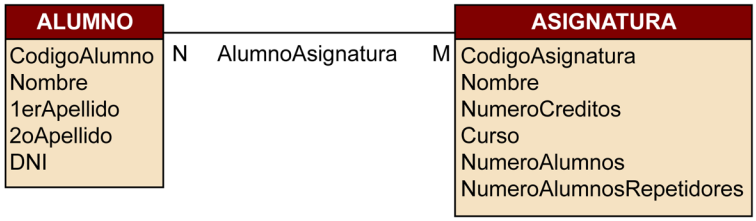
16. Indicad la única afirmación que es verdadera entre las cuatro siguientes:

a) El grado de una relación puede ser de una a una (1:1) o de una a dos (1:2).
b) El grado de una relación puede ser de una a muchas (1:n) o de dos a muchas (2:n).
c) El grado de una relación ternaria es siempre de una a tres (1:3).
d) Todas las afirmaciones anteriores son falsas.

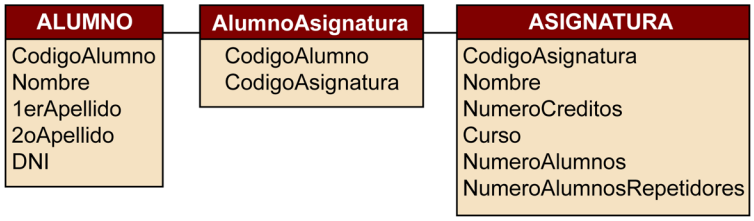
17. ¿Qué afirmación es verdadera?

a) Un atributo de una entidad en el modelo conceptual se convierte en un campo de una tabla en el modelo físico.
b) La relación n:1 indica que para un determinado registro de la primera tabla puede haber *n* registros en la segunda entidad que contengan el mismo atributo que es clave primaria en la primera tabla.
c) Las respuestas *a* y *b* son ciertas.
d) Todas son falsas.

18. En el modelo conceptual de una base de datos, se observan dos entidades con una relación *n:m*. Según la figura, cada alumno puede cursar unas o más asignaturas, y en cada asignatura, cursan uno o más alumnos. Los atributos que forman claves primarias aparecen subrayados.



Si se pasa al modelo relacional...



... indicad cuál de las siguientes afirmaciones referida a la tabla AlumnoAsignatura es falsa.

a) Se puede evitar que aparezca la tabla de relación AlumnoAsignatura que no existía en el modelo conceptual.
b) La relación entre ALUMNO y AlumnoAsignatura será *1:n*.
c) La clave primaria de la tabla ALUMNO, junto con la clave primaria de la tabla ASIGNATURA, forman la clave primaria de la tabla de relación AlumnoAsignatura.
d) Todas las afirmaciones anteriores son falsas.

19. Diseño de una base de datos

Os pedimos que diseñéis una pequeña base de datos que permita gestionar las actividades que se organizan en los diferentes espacios compartidos (sala de conferencias y aulas de formación) y las asociaciones que participan en un local de entidades.

El espacio dispone de tres espacios compartidos:

- la sala de conferencias,
- el aula 1, destinada a acciones de formación general,
- el aula 2, equipada con equipos informáticos, para poder llevar actuaciones de formación sobre las tecnologías de la información.

Aun así, estos espacios se pueden ampliar con el tiempo.

Se prevé disponer, por ejemplo, a medio plazo, de una tercera aula de formación y de una sala de exposiciones.

Estos espacios los pueden utilizar para las diferentes actividades que se organizan, tanto directamente por el local, como por asociaciones de la comarca.

Así pues, es necesario que la base de datos permita almacenar lo siguiente:

a) Lista de asociaciones, con sus principales datos (NIF, nombre entidad, responsable, dirección, teléfono de contacto, correo electrónico, web, etc.).
b) Relación de actividades organizadas (nombre actividad, responsable, fecha de inicio y fecha de finalización, asociación organizadora, etc.).
c) Espacios disponibles (identificador espacio, aforo máximo, etc.).
d) El uso de los espacios, por parte de las actividades, indicando los días y horarios.
e) Y la cesión, por otra parte, de los espacios a las asociaciones.

En el caso *d*, hay que considerar que una actividad puede utilizar más de un espacio, en diferentes días. Por ejemplo, un curso de gestión de entidades, que dura tres sábados y se lleva a cabo durante las mañanas, utiliza el primero y el segundo día el aula 1 y el tercer día, el aula 2, ya que se explican herramientas de apoyo a la gestión de entidades. En este caso, además, esta actividad no la organiza ninguna entidad, sino el mismo local.

Siempre hay que considerar el horario de uso, mañana o tarde.

Finalmente, en el caso *e* hay situaciones en las que una asociación pide el uso de un espacio para un día (horario de mañana o de tarde) con el fin de llevar a cabo alguna actuación interna o actividad que no se registra como tal.

En este caso, se habla de una cesión del espacio.

Por ejemplo, la Asociación Joven Cambra puede haber solicitado la cesión de la sala de conferencias para un viernes por la tarde, con el fin de hacer su asamblea general.

En estos casos, sólo nos interesa saber qué espacio ha sido cedido a la Asociación y para qué día y qué horario, pero no registramos la actividad.

Se pide, pues, que realicéis las actividades siguientes:

a) Diseño del **modelo conceptual de la base de datos** (modelo entidad-relación)
Identificación de las entidades, de sus atributos y de las interrelaciones entre ellas, con los atributos, si fuera el caso.
b) Generación del **modelo relacional de la base de datos**. Partiendo del modelo conceptual diseñado en el apartado 1, se debe convertir en una base de datos relacional, indicando las tablas correspondientes, los campos de cada una y su tipo, la clave primaria de cada tipo y las claves foráneas correspondientes.

Solucionario

Ejercicios de autoevaluación

1. La respuesta correcta es *d*. Todas las opciones son ciertas.

2. La respuesta correcta es *b*. En la memoria RAM se puede leer y escribir.

3. La respuesta correcta es *d*. Ninguna de las opciones anteriores es cierta.

4. La respuesta correcta es *b*. Los diferentes dispositivos de almacenamiento existentes ofrecen diferentes prestaciones y capacidades de almacenaje.

5. La respuesta correcta es la *c*. Dentro de un fichero, los registros se identifican por un único campo que es la clave.

6. La respuesta correcta es la *d*. Los ficheros de maniobras permiten ejecutar aplicaciones en las que los requerimientos de datos superan la capacidad de la memoria principal.

7. La respuesta correcta es la *a*. En la operación de fusión de ficheros se unen dos o más ficheros para integrarlos en uno solo.

8. La respuesta correcta es la *d*. Todas las afirmaciones anteriores son falsas.

9. La respuesta correcta es la *c*. El registro físico se corresponde con la cantidad de información que se puede leer y escribir al mismo tiempo en soporte físico.

10. La respuesta correcta es la *c*. Un sistema gestor de bases de datos (SGBD) accede a la información desde un lenguaje de alto nivel denominado *lenguaje de manipulación de datos* (DML).

11. La respuesta correcta es la *c*. Algunas veces las bases de datos presentan redundancias con el fin de mejorar los accesos y el rendimiento.

12. La respuesta correcta es la *c*. La utilización de una base de datos asegura la eliminación de las inconsistencias de datos.

13. La respuesta correcta es la *b*. Un buen sistema de base de datos debe permitir realizar cambios sobre el soporte físico de la base de datos sin que por ello se vean afectados los programas que la utilizan.

14. La respuesta correcta es la *d*. Todas las anteriores son falsas.

15. La respuesta correcta es la *b*. En el modelo conceptual, una entidad está formada por una serie de atributos.

16. La respuesta correcta es la *d*. Todas las afirmaciones anteriores son falsas.

17. La respuesta correcta es la *c*. Las opciones *a* y *b* son ciertas.

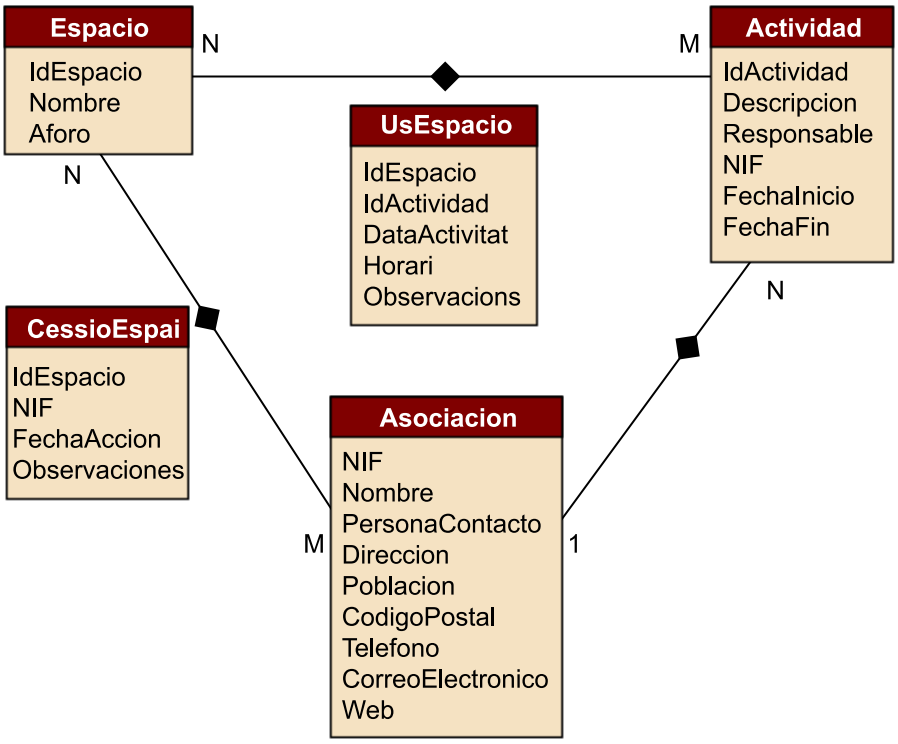
18. La respuesta correcta es la *a*. Se puede evitar que aparezca la tabla de relación "Alumno-Asignatura" que no teníamos en el modelo conceptual.

19. Diseño de una base de datos

Este ejercicio no tiene una única solución, ya que dependiendo del diseño del modelo conceptual, la base de datos relacional final será una u otra.

En una situación real, durante el diseño de la base de datos, el responsable solicita diferentes requerimientos al "cliente" para ir determinando posibles ambigüedades y dudas. En vuestro caso, el consultor de la asignatura actúa como cliente.

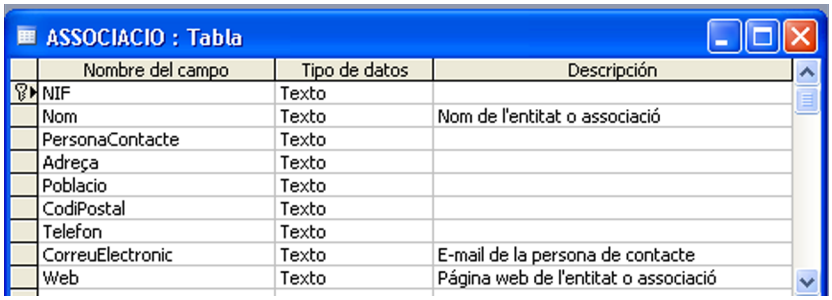
a) Diseño del modelo conceptual (entidades e interrelaciones, con sus atributos)



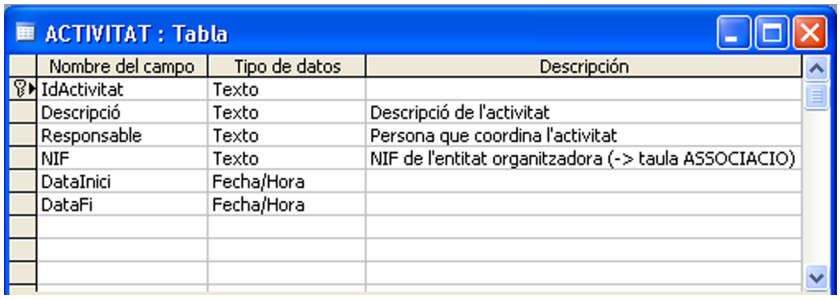
Las entidades se muestran en rectángulos con su nombre en la cabecera y los atributos a continuación. Las relaciones entre entidades se muestran con un rombo junto al que aparece su nombre, seguido de sus atributos. La cardinalidad de las relaciones se indica mediante 1, N o M al lado de las entidades que participan. Los atributos que forman clave primaria están subrayados.

b) Generación del modelo relacional (con MS Access)

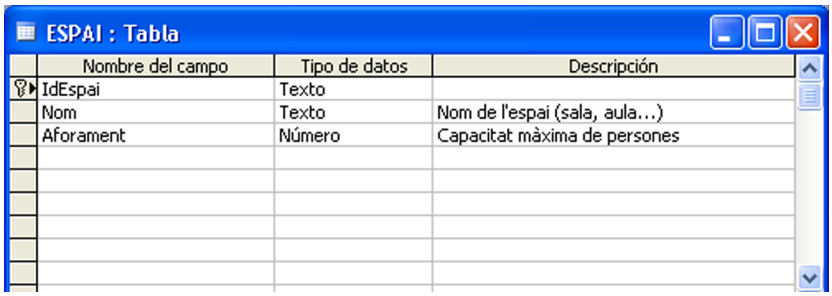
- Diseño de las tablas correspondientes a las entidades del modelo conceptual: ASOCIACIÓN, ACTIVIDAD y ESPACIO. Cada tabla incluye la definición de los campos, los tipos de datos de cada uno de ellos y la identificación de su clave primaria (en Access, los campos que la forman se distinguen con el símbolo de una clave).



Nombre del campo	Tipo de datos	Descripción
NIF	Texto	
Nom	Texto	Nom de l'entitat o associació
PersonaContacte	Texto	
Adreça	Texto	
Poblacio	Texto	
CodiPostal	Texto	
Telefon	Texto	
CorreuElectronic	Texto	E-mail de la persona de contacte
Web	Texto	Página web de l'entitat o associació



Nombre del campo	Tipo de datos	Descripción
IdActivitat	Texto	
Descripció	Texto	Descripció de l'activitat
Responsable	Texto	Persona que coordina l'activitat
NIF	Texto	NIF de l'entitat organitzadora (-> taula ASSOCIACIO)
DataInici	Fecha/Hora	
DataFi	Fecha/Hora	



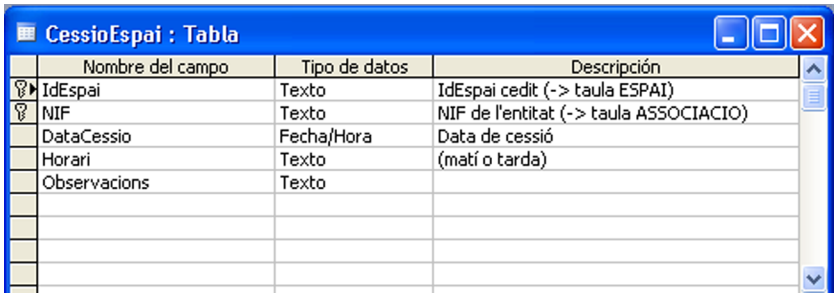
ESPAI : Tabla

Nombre del campo	Tipo de datos	Descripción
IdEspai	Texto	
Nom	Texto	Nom de l'espai (sala, aula...)
Aforament	Número	Capacitat màxima de persones

- Diseño de las tablas intermedias generadas a partir de las relaciones M:N del modelo conceptual: UsEspacio y CessioEspacio.

UsEspai : Tabla

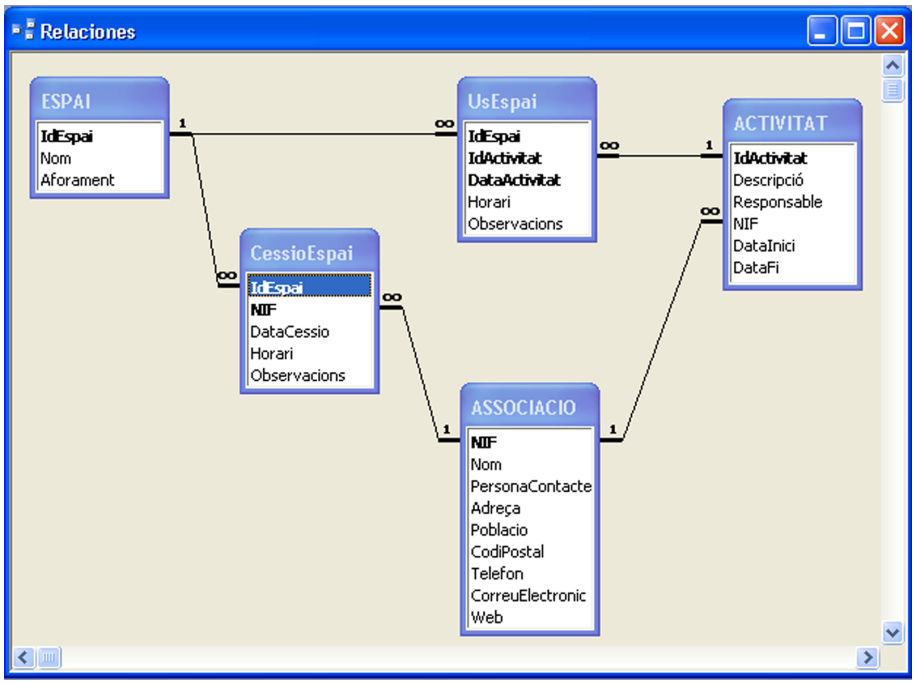
Nombre del campo	Tipo de datos	Descripción
IdEspai	Texto	IdEspai utilitzat (-> taula ESPAI)
IdActivitat	Texto	IdActivitat realitzada (-> taula ACTIVITAT)
DataActivitat	Fecha/Hora	Data d'utilització de l'espai
Horari	Texto	(matí o tarda)
Observacions	Texto	



CessioEspai : Tabla

Nombre del campo	Tipo de datos	Descripción
IdEspai	Texto	IdEspai cedit (-> taula ESPAI)
NIF	Texto	NIF de l'entitat (-> taula ASSOCIACIO)
DataCessio	Fecha/Hora	Data de cessió
Horari	Texto	(matí o tarda)
Observacions	Texto	

- La relación entre las entidades ASOCIACIÓN y ACTIVIDAD no requiere de una tabla intermedia, ya que es de tipo uno a muchos (*1:n*). Es decir, una asociación puede organizar distintas actividades, pero una actividad sólo la organiza una asociación.
- Implementación de las relaciones entre las tablas de la BD.



Los conjuntos de campos que forman las claves primarias se distinguen en negrita.

Los campos que son clave foránea aparecen junto al símbolo de infinito (¥).

Bibliografía

Silberschatz, A.; Korth, H.L.; Sudarsahn, S. (1998). *Fonaments de Bases de Dades (3.)*. Madrid: McGraw - Hill.

Ullman, J. (1991). *Principles of Database and knowledge - Base Systems*. Computer Science Press.

de Miguel, A.; Piattini, M. (1999). *Fonaments i models de bases de dades (2)*. Madrid : RA-DT.

Presuman, Roger S. (1997). *Ingeniería de Software. Un enfoque práctico*. Madrid: Mc Graw Hill

Software

Ramon Costa i Pujol
Jeroni Vivó i Plans

PID_00153112

Universitat Oberta de Catalunya
www.uoc.edu

Índice

Introducción

El software de un sistema informático está constituido por el conjunto de programas ejecutables en aquel sistema.

Los sistemas informáticos, es decir, la combinación del hardware, como son los ordenadores, la infraestructura de las redes y los sistemas de comunicación (cableado, etc.), con el software, han permitido mejorar muchos aspectos de la sociedad, ya que la aplicación de estos sistemas engloba todos los ámbitos, tanto genéricos como, por ejemplo, de:

- Negocios.
- Medicina.
- Educación.
- Ciencia y tecnología.
- Ingeniería y arquitectura.

O si se trata de ámbitos propios de las unidades funcionales o departamentales de las organizaciones, como, por ejemplo:

- Producción.
- Logística y distribución.
- Atención al cliente.
- Formación y gestión de los RRHH.
- Gestión de la planificación.

Todo este conjunto de software se desarrolla utilizando unos lenguajes de programación y siguiendo unas pautas y metodologías conocidas como *ingeniería del software*.

La construcción de programas de ordenador utilizando directamente el lenguaje nativo de la máquina, tal como se hacía en los inicios de la informática, tiene muchos inconvenientes. La única manera de superar estos inconve-

nientes consiste en desarrollar lenguajes adecuados para programar cualquier ordenador y que se puedan traducir al lenguaje que entienda una máquina concreta.

Uno de los apartados de este módulo consistirá en analizar qué caracteriza un lenguaje de programación.

Un lenguaje de programación

Un lenguaje de programación es un conjunto de normas sintácticas englobadas bajo un nombre (como puede ser C, Pascal, Java, etc.) que especifican cómo y cuándo se puede utilizar un conjunto de instrucciones determinado.

A partir de los lenguajes de programación se pueden diseñar programas, que después se convierten a códigos máquina por parte del compilador correspondiente del lenguaje para su ejecución, y ofrecer una visión general de cómo se produce la traducción de programas escritos en un lenguaje de alto nivel al lenguaje máquina de un ordenador.

No pretendemos, en esta asignatura, conocer con detalle los lenguajes de programación, sino ofrecer una visión abstracta de estos lenguajes que ayude a la comprensión de sus aspectos más interesantes (como los ámbitos de aplicación).

El diseño, el desarrollo y la implantación de un sistema informático engloba tanto los aspectos de software (diseño y desarrollo de aplicaciones, parametrización de programas existentes, etc.), como de hardware (instalación y configuración de ordenadores y redes, parametrización de servidores de correo, de ficheros, de red, etc.).

La gestión de todas estas tareas se realiza siguiendo los parámetros de la gestión de proyectos.

Una parte importante de todo sistema informático es la adquisición del software requerido para implantar la solución pedida (servidores de redes, de ficheros, programas de desarrollo, aplicaciones ya hechas, etc.).

Implantar, poner en marcha una instalación informática

También se utiliza esta palabra para referirse a la fase dentro del ciclo de vida de un programa durante la cual se codifica en un lenguaje de programación determinado.

Esta adquisición del programa se debe realizar atendiendo unas políticas de compra y uso de licencias marcadas por los fabricantes.

En definitiva, el objetivo de este módulo es introducir al estudiante en los conceptos principales:

- del software y sus aplicaciones,
- el desarrollo del software y los lenguajes de programación,
- la gestión de proyectos,
- el uso de las licencias de software.

Objetivos

Los **objetivos generales** que el estudiante puede alcanzar son los siguientes:

1. Conocer el concepto de software y sus principales aplicaciones.
2. Conocer las principales herramientas disponibles para el desarrollo de software, en especial por lo que respecta a lenguajes de programación y a gestión de proyectos.
3. Conocer las tipologías de licencias de software y sus implicaciones prácticas.

Estos objetivos generales se desglosan en los **objetivos específicos** siguientes:

1. Definir el concepto de software y enumerar sus principales características.
2. Enumerar las principales aplicaciones del software en función de su finalidad.
3. Describir el concepto de desarrollo de software para crear programas y aplicaciones.
4. Enumerar las principales características y diferencias entre los lenguajes máquina, ensamblador y de alto nivel.
5. Conocer un conjunto de los principales lenguajes de programación y sus diferencias.
6. Describir las principales características de una herramienta CASE.
7. Conocer los principales conceptos y elementos de la ingeniería del software.
8. Identificar los principales conceptos de la gestión de proyectos en general y enumerar las principales fases.
9. Describir los principales conceptos de la planificación de un proyecto y su seguimiento: tareas, recursos, herramientas y técnicas.
10. Identificar las principales características de un proyecto informático y enumerar sus principales fases y tareas.

11. Identificar y enumerar los principales papeles de los miembros de un equipo de trabajo en un proyecto informático.

12. Describir el concepto de licencias de software.

13. Listar diferentes posibilidades y opciones en la compra y uso de licencias de software.

14. Enumerar las principales características de las licencias "*Freeware*" y "*Shareware*".

1. Software: una visión general

El software de un sistema informático, más conocido como software, está constituido por el conjunto de programas ejecutables y ficheros necesarios por su funcionamiento.

Dentro del software, se incluyen herramientas tan variadas como:

- El sistema operativo.
- Las interfaces de usuario (sistema de ventanas, por ejemplo).
- Lenguajes de programación (Pascal o el entorno Visual Studio, por ejemplo).
- Herramientas o utilidades (compresores de ficheros, antivirus).
- Aplicaciones de cualquier especialidad y tipo (procesadores de texto, herramientas de diseño y gestión de imágenes).
- Ficheros de datos y bases de datos con los que trabajan estos programas; contienen los datos y las estructuras de datos necesarios para el funcionamiento del software y las generadas por su propio uso.

Una parte importante del software de un sistema lo forman las herramientas o utilidades. Dentro de este paquete de programas, se puede incluir software tan variado como programas de hoja de cálculo, tratamiento de textos como aplicaciones de uso común, los gestores de base de datos, los paquetes de todo en uno (herramientas conocidas como de productividad personal), los paquetes de gestión, así como otros paquetes de utilidades, ya mencionados anteriormente.

El **software** es todo aquel conjunto de programas o utilidades que se ejecutan en un ordenador.

Son muchas las empresas que se dedican a desarrollar este tipo de aplicaciones. Se trata de un mercado muy dinámico y con una competencia feroz, lo que provoca que la aparición de nuevas versiones y revisiones sea constante. Estas nuevas versiones quieren corregir errores antiguos y, a menudo, también presentan nuevas funcionalidades, muchas de ellas demandadas por los propios usuarios.

Los computadores tienen la capacidad de realizar muchas y distintas tareas, siempre y cuando tengan el software adecuado.

Los ordenadores permiten realizar trabajos que antes necesitaban un personal muy especializado en varios campos (delineación, analistas financieros, programadores) para poder llevarlos a cabo, facilitando la automatización y simplificación de las tareas de soporte de estos profesionales. Los software no pueden sustituir el conocimiento de los delineantes, analistas financieros o programadores; lo que pueden hacer es automatizar procesos rutinarios y dar elementos para facilitar el resto, de manera que el trabajo se haga de una forma más eficiente.

En la actualidad, la gran mayoría de estos trabajos se pueden realizar más fácilmente mediante un ordenador personal, el software adecuado y una mínima formación en el uso y el trabajo de este software (siempre que el usuario tenga un nivel de competencia adecuado en el ámbito temático de utilización).

Además, el software de inteligencia artificial permite ir más allá de la simple automatización a la hora de ayudar a resolver problemas complejos.

Una primera clasificación del software permitiría diferenciar dos grandes categorías: el software de sistema y el software de aplicación. Es decir, las aplicaciones y utilidades que sirven para controlar el hardware (sistemas operativos, servidores de bases de datos, servidores de correo electrónico, servidores de ficheros e impresoras, etc.) y los programas que trabajan en este software y hardware (aplicaciones empresariales, personales, etc.).

Una posible organización del software podría ser distinguir entre el software de sistema y el software de aplicación.

En esta unidad trataremos de las diferentes aplicaciones del software, proponiendo una clasificación posible de estos programas.

1.1. Aplicaciones del software

El software se puede aplicar a cualquier situación en la que se haya definido previamente un conjunto específico de pasos procedimentales, lo que se conoce como *algoritmos* (un problema determinado se intenta descomponer en un conjunto de problemas menores, lo cual permite llegar a crear algoritmos muy potentes).

Con el fin de determinar la naturaleza del software, hay que tener en cuenta tanto el contenido como la información gestionada.

Por una parte, el contenido tiene que ver con el significado y la forma de la información de entrada y de salida.

Aplicaciones bancarias

Por ejemplo, muchas aplicaciones bancarias utilizan unos datos de entrada muy estructurados (una base de datos) y producen "informes" con formatos determinados.

El software que controla una máquina automática (por ejemplo, un control numérico) actúa sobre elementos de datos discretos con una estructura limitada y produce órdenes concretas para la máquina en una rápida sucesión.

Por otra parte, en cambio, la gestión de la información también ha de tener en cuenta la predictibilidad de la orden y del tiempo de llegada de los datos.

Un programa de ingeniería acepta datos que están en un orden predefinido, ejecuta el algoritmo sin interrupción y produce datos resultantes en un informe o formato gráfico.

Programa de ingeniería

Ejemplos de este tipo de aplicaciones son:

- Consolidación de movimientos mensuales de los pedidos de una empresa.
- Gestión de las nóminas.

En cambio, un sistema operativo multiusuario recibe entradas que tienen un contenido variado o que se producen en instantes arbitrarios, ejecuta algoritmos que se pueden interrumpir por condiciones externas y produce una salida que depende de una función del entorno y del tiempo. Las aplicaciones con estas características se denominan indeterminadas.

Aplicaciones indeterminadas

Otros ejemplos de aplicaciones indeterminadas serían:

- Sistemas de monitoreo de plantas industriales.
- Aplicaciones de control de alarmas y seguridad en un edificio.

Con el fin de determinar la naturaleza del software, hay que tener en cuenta tanto el contenido, como la información que gestiona.

Es muy difícil establecer categorías genéricas para las aplicaciones del software. A medida que aumenta la complejidad del software, se hace más complicado establecer fronteras nítidamente separadas.

Con el fin de repasar las posibles aplicaciones del software, se ha decidido categorizarlas en los apartados siguientes:

- Software de sistemas.
- Software de tiempo real.
- Herramientas de uso personal.
- Software de ingeniería y científico.
- Herramientas de gestión y rediseño de procesos organizativos.

- Aplicaciones de inteligencia artificial.

Se habría podido clasificar el software según otras taxonomías o detallar más esta organización (gestión del conocimiento, control de la producción, sistemas de gestión de información gráfica, etc.) o por su aplicación en el ámbito de Internet y las relaciones entre empresas y usuarios (B2B[1], B2C[2], etc.), pero se ha optado por introducir, en este módulo, estos seis tipos de aplicaciones.

(1)B2B es el acrónimo de las palabras inglesas *business to business* ('negocio a negocio'). Esta abreviatura se utiliza mucho en el ciberespacio para referirse a todas las relaciones comerciales que establecen las empresas entre sí por medio de Internet.

(2)B2C es el acrónimo de las palabras inglesas *business to consumer* ('negocio a consumidor'). Esta abreviatura se utiliza mucho en el ciberespacio para referirse a las relaciones comerciales que establecen las empresas con los consumidores finales por Internet.

1.1.1. Software de sistemas

Son un conjunto de programas que se han escrito para servir a otros programas.

Algunos programas de sistemas como los compiladores o los editores y las utilidades de gestión de archivos procesan estructuras de información complejas pero determinadas.

Otras aplicaciones de sistemas son, por ejemplo, determinados componentes del sistema operativo, utilidades de gestión de los periféricos o procesadores, de telecomunicaciones que procesan datos muy indeterminados.

En cualquier caso, el área del software de sistemas se caracteriza por:

- Una fuerte interacción con el hardware de la computadora.
- Concurrencia de usuarios.

Estos dos elementos combinados hacen que este tipo de software tenga que prever el compartimento de recursos y haya de hacer, a la vez, una gestión de los procesos muy cuidadosa para dejar satisfechos a todos los usuarios, entendiendo como usuarios los programas que se ejecutan por encima del software de sistema.

1.1.2. Software de tiempo real

El software que mide, analiza y controla sucesos del mundo real a medida que ocurren se denomina *de tiempo real*.

Entre los elementos del software de tiempo real se incluyen:

- Un componente de adquisición de datos que recoge y da formato a la información recibida del entorno externo.
- Un componente de análisis que transforma la información según lo requiera la aplicación.
- Un componente de control/salida que responda al entorno externo.
- Un componente de monitoreo que coordina todos los demás componentes, de manera que se pueda mantener la respuesta en tiempo real (típicamente en el rango de un milisegundo a un minuto).

Cabe tener en cuenta que el concepto "tiempo real" significa que el sistema ha de responder dentro de unos vínculos estrictos de tiempo para evitar que se produzca algún desastre.

Ejemplos de software de tiempo real

Algunos ejemplos son los sistemas de monitoreo de una planta industrial que controlan todas las máquinas que intervienen en la cadena de producción, o el sistema de control de las alarmas de un edificio o de una planta nuclear o, más comunes para el usuario de a pie, el software de un cajero automático o el de un peaje.

1.1.3. Herramientas de uso personal

Desde la aparición de los ordenadores personales (PC) a finales de la década de los setenta e inicios de los ochenta se han desarrollado aplicaciones y programas pensados y diseñados para ayudar al trabajo personal y las tareas cotidianas, tanto en la oficina como en el hogar:

- Procesadores de textos.
- Hojas de cálculo.
- Gráficas por ordenador.
- Reproductores multimedia de vídeo y sonido.
- Juegos y aplicaciones de entretenimiento.
- Gestión de agendas personales.
- Aplicaciones financieras, de negocios y personal.

Éstas, y muchas más, son algunas de los centenares de aplicaciones diseñadas para ayudar a las personas en sus tareas cotidianas.

1.1.4. Software de ingeniería y científico

El software de ingeniería y científico está caracterizado por los algoritmos optimizados para la manipulación matemática de los datos.

Las aplicaciones de esta tipología van desde:

- Soporte a la observación astronómica.
- Predicción meteorológica.
- Monitorización de procesos industriales.
- Aplicaciones de soporte a la diagnosis médica.
- Aplicaciones de soporte al I+D.
- Etc.

O por aplicaciones más próximas a la gente de la calle como:

- El control y sincronización de semáforos.
- Cálculo de estructuras de edificios.
- Control del flujo de la corriente eléctrica.
- Etc.

En los últimos años, las aplicaciones de ingeniería/ciencia se han alejado de los algoritmos convencionales numéricos y el diseño asistido por ordenador (del inglés CAD), la simulación de sistemas u otras aplicaciones interactivas también tienen características del software de tiempo real e, incluso, del software de sistemas. En estos momentos muchas aplicaciones científicas permiten tratar, en tiempo real, medidas y datos como, por ejemplo, el seguimiento de la situación de los elementos meteorológicos.

1.1.5. Herramientas de gestión y rediseño de procesos organizativos

La gestión de información comercial y corporativa constituye la mayor de las áreas de aplicación del software.

Sistemas como la gestión de nóminas, la compatibilidad, la gestión de inventarios, etc. han evolucionado hacia el software de sistemas de información (SI), que accede a una o más bases de datos grandes que contienen información de la organización.

Las aplicaciones en esta área reestructuran los datos existentes para facilitar las operaciones comerciales o gestionar la toma de decisiones. Además de las tareas convencionales de procesamiento de datos, las aplicaciones de software de gestión también realizan cálculos interactivos (por ejemplo, el procesamiento de transacciones en puntos de venta).

La evolución y la aplicación de este software ha ido dando apoyo, cada vez más, a otras funciones de la organización y sus departamentos, por lo que se han convertido en herramientas de gestión de la mayor parte de los procesos de las compañías:

- el proceso comercial,
- el proceso de compras,
- las operaciones de producción y fabricación,
- la gestión de los recursos humanos.

Para estas tareas han aparecido soluciones específicas como:

- SCM (*supply chain management*, 'gestión de la cadena de suministro'),
- CRM (*customer relationship management*, 'gestión de las relaciones con el cliente'),
- BPR (*business process reengineering*, 'reingeniería de procesos de negocio'),
- e-learning (formación por medio de herramientas telemáticas),
- e-rrhh (soluciones de apoyo a la gestión del departamento de recursos humanos),
- *Content management* ('gestión de contenidos y documentos'),
- *Knowlegde management* ('gestión del conocimiento').

1.1.6. Aplicaciones de inteligencia artificial

El software de inteligencia artificial (IA) utiliza algoritmos no numéricos para resolver problemas complejos para los cuales no es adecuado el cálculo o el análisis directo.

Diferentes áreas de la IA son los sistemas expertos, también denominados sistemas basados en el conocimiento: reconocimiento de patrones, reconocimiento de imágenes, de voz y de escritura.

Hay diferentes formas de implementar aplicaciones de inteligencia artificial; la que más se aproxima al funcionamiento del cerebro humano y de los animales son los algoritmos que implementan redes neuronales que simulan, hasta donde la tecnología y los conocimientos humanos permiten, la estructura de procesos del cerebro (las funciones de la neurona biológica) y, a la larga, pueden llevar a una clase de software que reconozca patrones complejos y a aprender "de experiencias" pasadas.

Las aplicaciones de inteligencia artificial se usan en ámbitos como la ingeniería, medicina, militar y la economía. También se han usado para juegos de estrategia, como el ajedrez, y para hacer videojuegos.

2. Desarrollo de software

Todo software debe haber sido diseñado y haber sido creado, o como se dice en términos informáticos, desarrollado.

Para desarrollar estos programas y aplicaciones, los "programadores" disponen de un conjunto de lenguajes de programación o desarrollo.

Existe una gran cantidad de lenguajes de desarrollo. Algunos se han diseñado para crear programas de un tipo específico o para un tipo de aplicaciones concretas (aplicaciones científicas y técnicas, inteligencia artificial, gestión, etc.) y otros son de aplicación "general".

Los programas desarrollados en estos lenguajes se deben traducir, posteriormente, al código fuente (un lenguaje inteligible por el ordenador y su procesador). Hay dos tipos de traductores, los compiladores y los intérpretes.

Con el fin de optimizar este proceso de desarrollo y minimizar, por otra parte, los posibles errores en la producción del software, se ha desarrollado lo que se denomina *ingeniería del software*, que engloba un conjunto de métodos, herramientas y procedimientos que facilitan el trabajo a los programadores.

Los objetivos de este apartado son:

- Introducir al alumno en los principales conceptos de los lenguajes de programación.
- Hacer un repaso de las "generaciones" de lenguajes.
- Enumerar los principales lenguajes de desarrollo existentes y su aplicación.
- Indicar las principales diferencias entre el lenguaje máquina, el lenguaje ensamblador y los lenguajes de alto nivel y sus características.
- Introducir las herramientas CASE.
- Enumerar los principales conceptos de la ingeniería del software.

2.1. Lenguajes de programación

Los desarrolladores de software, o "programadores", utilizan lenguajes de programación para crear y desarrollar los paquetes de software o programas como, por ejemplo, los procesadores de textos, las hojas de cálculo o los sistemas de mensajería o las aplicaciones web que los diferentes usuarios utilizan cuando trabajan con sus ordenadores.

Toda aplicación informática ha sido desarrollada utilizando un lenguaje de programación.

Se componen de los siguientes elementos:

- un léxico, que es el conjunto de símbolos permitidos o vocabulario,
- una sintaxis, que son las reglas que indican cómo realizar las instrucciones del lenguaje,
- y una semántica, que corresponde a las reglas que permiten determinar el significado de cualquier construcción del lenguaje.

Los programas que puede ejecutar el ordenador se han de "redactar" en el lenguaje nativo del procesador.

Es decir, cada instrucción debe estar en código binario y directamente relacionada con los circuitos del procesador.

Ahora bien, programar instrucciones completamente en código binario, es un proceso demasiado lento y sujeto a errores.

Por ello, los lenguajes de programación se han diseñado para permitir que el desarrollador escriba las instrucciones parecidas a un lenguaje humano, casi siempre en inglés.

Estas instrucciones se convierten, después, en el código binario correspondiente mediante unos programas denominados *compiladores*.

Tal como veremos más adelante, los compiladores traducen las instrucciones de un lenguaje de programación de alto nivel a código binario de bajo nivel, como una persona podría traducir de un idioma a otro, por ejemplo, del catalán al castellano.

De esta manera, un compilador o traductor convierte los programas que el desarrollador ha programado utilizando un lenguaje de alto nivel a un código fuente y un código objeto.

El código fuente es el conjunto de instrucciones escritas en un lenguaje de programación, mientras que el código objeto es el conjunto de instrucciones binarias que puede ejecutar el ordenador.

Los programas que el ordenador puede ejecutar han de estar "redactados" en el lenguaje nativo de sus procesadores. Por ello, los programas desarrollados utilizando los lenguajes de programación de alto nivel se han de traducir a un código fuente y un código objeto, inteligibles para el procesador del ordenador.

2.1.1. Lenguajes máquina, ensamblador y alto nivel

En este apartado, se profundiza en las principales características del lenguaje máquina, ensamblador y los de alto nivel.

Lenguaje máquina

Tal como se ha mencionado en la introducción, el lenguaje máquina es el único lenguaje que entiende directamente el ordenador, o mejor dicho, el procesador, ya que su estructura está totalmente adaptada a los circuitos de la máquina.

Por lo tanto, su forma de expresión está muy alejada de la comprensión y el análisis de las personas.

La programación con este tipo de lenguajes es muy complicada, ya que requiere un conocimiento amplio de la arquitectura física del ordenador por parte del programador.

Por contra, el código máquina permite que el desarrollador pueda utilizar la totalidad de los recursos que ofrece el ordenador, de manera que se pueden obtener programas muy eficientes, tanto en tiempo de ejecución como en ocupación de memoria.

Estos lenguajes constituyen la primera generación de lenguajes de programación.

Sus principales características son las siguientes:

- Las instrucciones se expresan en código binario, codificadas como cadenas de ceros (0) y unos (1), lo que hace que sea muy difícil de entender y modificar (mantener).

- Los datos se referencian por medio de las direcciones de memoria donde se encuentran y no por nombres de variables o constantes como pasa en los lenguajes de alto nivel.

- Cada una de las instrucciones realiza operaciones muy simples, de manera que el programador ha de saber cómo combinar el número reducido de instrucciones de las que dispone con el fin de realizar los algoritmos buscados.

- Las instrucciones tienen un formato muy rígido con respecto a la posición de los diferentes campos (por ejemplo, CÓDIGO OPERACIÓN - PRIMER OPERANDO - SEGUNDO OPERANDO), por lo que la redacción de las instrucciones es muy poco flexible.

- Tal como ya hemos mencionado, el lenguaje máquina o lenguaje nativo depende de la CPU (procesador) de cada ordenador y está ligado a ella, por lo que los programas en código máquina no se pueden transferir de un modelo de ordenador a otro y sólo pueden ejecutarse en el procesador con el cual está vinculado el código máquina correspondiente.

- En un programa escrito en lenguaje máquina no se pueden incluir comentarios que ayuden a su seguimiento y comprensión, y que faciliten la modificación posterior (mantenimiento).

El lenguaje máquina es lo único que entiende directamente el ordenador (su procesador). La programación con este tipo de lenguajes es muy complicada, pero permite que el desarrollador pueda utilizar la totalidad de los recursos que ofrece el ordenador, obteniendo programas muy eficientes, tanto en tiempo de ejecución como en ocupación de memoria.

Lenguaje ensamblador

La mayoría de las limitaciones comentadas de los lenguajes máquina las resuelve, parcialmente, el lenguaje ensamblador y casi totalmente los lenguajes de alto nivel o también conocidos como *simbólicos*.

Para evitar que los programadores deban programar directamente en código binario o máquina, se desarrollaron unos programas que permitieran traducir las instrucciones a código máquina.

Estos programas son la segunda generación de lenguajes de programación y se denominaron *ensambladores*.

Podían leer las instrucciones que las personas "podían entender" mejor, redactadas en lenguaje ensamblador, y las convertían en una instrucción de lenguaje máquina.

Aun así, este lenguaje también es de bajo nivel, ya que cada una de las instrucciones corresponde a una instrucción del lenguaje máquina.

Cada procesador tiene, por lo tanto, su lenguaje ensamblador, que traduce el código fuente, línea por línea, a código de máquina y crea el fichero ejecutable de la aplicación.

Código fuente

Listado de texto que contiene las instrucciones que componen una aplicación en un determinado lenguaje de programación que una vez compilado o parecido con la correspondiente herramienta da lugar a un programa funcional ejecutable bajo un sistema operativo.

Como principales características se puede enumerar que:

- Estos lenguajes utilizan una notación simbólica o mnemotécnica para representar el código de operación, lo que evita los código numéricos tan difíciles de utilizar. Estos códigos mnemotécnicos están constituidos por abreviaturas de las operaciones en inglés. Por ejemplo, la instrucción de sumar se representa en la mayoría de los lenguajes ensamblador con las letras ADD.

- En lugar de utilizar direcciones binarias absolutas en la memoria, los datos se pueden identificar por nombres, lo que se conoce como *dirección simbólica*.

- El programador puede utilizar comentarios entre las líneas de instrucciones de los programas, lo que los convierte en más inteligibles y simplifica su seguimiento y posterior modificación.

- El lenguaje ensamblador continúa siendo muy importante, ya que ofrece al programador el control total de la máquina y, por lo tanto, le permite generar un código compacto, rápido y eficiente.

A pesar de todo, el lenguaje ensamblador presenta la mayoría de los inconvenientes que también tiene el lenguaje máquina, como, por ejemplo:

- Un repertorio muy reducido de instrucciones.

- Un formato muy rígido para las instrucciones.

- La baja portabilidad entre procesadores y su fuerte dependencia del hardware.

Para solucionar de alguna manera la limitación que implica disponer de un repertorio de instrucciones tan reducido, se han desarrollado unos ensambladores especiales denominados *macroensambladores*.

Los lenguajes que traducen los macroensambladors tienen "macroinstrucciones", cuya traducción da lugar a varias instrucciones máquina y no sólo a una única.

Ejemplos tipos macroinstrucciones

Por ejemplo, hay macroinstrucciones para multiplicar, dividir, transferir bloques de memoria principal a disco, etc.

Debido a estas limitaciones los programador no suelen desarrollar los programas en lenguaje ensamblador, sino que lo hacen con lenguajes de más alto nivel.

Los lenguajes ensambladores se definieron para evitar que los programadores hubieran de programar directamente en código binario o máquina. Cada procesador tiene su propio lenguaje ensamblador que traduce el código fuente, línea por línea, a código máquina, lo que hace que también sean unos lenguajes de bajo nivel.

Lenguajes de alto nivel

La tercera generación de lenguajes de programación tenía como objetivo facilitar la tarea del desarrollador al permitir crear programas independientes de la máquina utilizando una sintaxis muy parecida al inglés.

Ejemplos de lenguajes

Ejemplos de estos lenguajes son el Cobol, el Basic, el Fortran, el C o el Pascal.

Con estos lenguajes los programadores pueden escribir en una sola instrucción el equivalente a varias instrucciones complicadas de bajo nivel.

Para saber más

Para más información sobre lenguajes de programación, podéis ver también el apartado "Principales lenguajes de desarrollo" de este mismo módulo.

Tienen unas instrucciones más fáciles de entender y proporcionan facilidades para expresar alteraciones del flujo de control de una manera bastante intuitiva.

Ejemplo de rutina de código

Por ejemplo, una rutina de código desarrollada en un lenguaje de alto nivel tendría la estructura siguiente:

```
A= 0

FOR i = 1 TO 10

A = A + (i * i)

NEXT i
```

En este caso, esta rutina calcula la suma de los primeros 10 cuadrados de los números naturales (1 * 1 + 2 * 2 + 3 * 3 + ... + 10 * 10)

Independientemente del lenguaje de alto nivel en el que se escriba el programa, lo debe traducir un compilador al lenguaje máquina para que lo pueda ejecutar el procesador correspondiente.

Sus características principales son las siguientes:

- Son independientes de la arquitectura física del ordenador, por lo cual se pueden utilizar los mismos programas en ordenadores de arquitecturas diferentes, sin necesidad de conocer el hardware específico de la máquina. Sólo es necesario "compilar"[3]el programa con el compilador correspondiente.

Para saber más

Para más información sobre los compiladores, podéis ver el apartado "compiladores" de este mismo módulo.

(3)"Compilar"

Acción de traducir código escrito en formato ASCII a lenguaje máquina (código máquina) mediante un compilador.

- Como ya hemos mencionado, la ejecución de un programa en un lenguaje de alto nivel requiere una traducción al lenguaje máquina del ordenador en el que se ejecutará. Una sentencia en un lenguaje de alto nivel corresponde, una vez traducida, a varias instrucciones en lenguaje máquina.

- Utilizan notaciones próximas a las utilizadas por las personas, con lo cual se pretende una aproximación mayor al lenguaje natural o algebraico, de manera que las instrucciones están expresadas en texto y es posible introducir comentarios en las líneas de código de los programas y su escritura está, normalmente, basada en reglas parecidas a las humanas.

- Ofrecen instrucciones potentes como, por ejemplo, funciones matemáticas (seno, coseno, conversión de enteros a reales, etc.), operadores específicos de entrada o salida como, por ejemplo, "PRINT" u operadores de tratamiento de cadenas de caracteres como, por ejemplo, "extraer un conjunto de caracteres de una línea de texto" o "buscar un subtexto en una cadena". También ofrecen la posibilidad de crear tus propias funciones.

- A diferencia de los lenguajes máquina y ensamblador, éstos son menos eficientes.

Todas estas características ponen de manifiesto un acercamiento a las personas y un alejamiento de las máquinas.

Por ello, todos los programas escritos en lenguaje de alto nivel no se pueden tratar directamente por el ordenador y es necesario traducir al lenguaje máquina para ser ejecutado.

Estos programas que realizan estas traducciones se denominan traductores y han sido desarrollados para cada ordenador específico.

Una evolución de estos lenguajes fueron los enmarcados dentro de la cuarta generación (4GL), que permiten generar de manera automática la mayor parte de los procedimientos (rutinas o conjuntos de instrucciones de código en lenguaje de alto nivel) de un programa. De esta manera, el desarrollador sólo debe indicar qué quiere hacer, pero no cómo.

Un programador que trabaje con un lenguaje de tercer nivel como, por ejemplo, el C, escribe instrucciones de lo que se ha de hacer e indica cómo hacerlo por medio de los algoritmos redactados.

En cambio, con los lenguajes 4GL (que permiten generar código sin tener que desarrollarlo, a partir de especificaciones), los desarrolladores o usuarios (analistas funcionales, expertos no técnicos) pueden escribir programas de manera sencilla, por ejemplo, para consultar una base de datos o para crear sistemas de información personal o departamentales.

Muchos de estos lenguajes disponen de una interfaz gráfica.

Estos lenguajes convierten las especificaciones indicadas por el desarrollador o bien en lenguajes de tercer nivel, que el mismo programador puede refinar, o bien generan las instrucciones en código máquina directamente.

Ejemplos de lenguajes 4GL son la mayoría de las herramientas CASE de diseño de bases de datos, modelización de procesos, etc.

Finalmente, hay que indicar que la quinta generación de los lenguajes de programación son los denominados *lenguajes naturales*, que si bien están en sus inicios, facilitan una aproximación total al lenguaje humano.

Con los lenguajes de alto nivel, los programadores pueden escribir en una sola instrucción el equivalente a varias instrucciones complicadas de bajo nivel. Ahora bien, estos programas han de ser, posteriormente, traducidos a un lenguaje inteligible por el procesador.

2.1.2. Traductores

Tal como ya se ha comentado anteriormente, dado que un ordenador sólo puede interpretar y ejecutar el código máquina, existen unos programas especiales, denominados *traductores*, que traducen programas escritos en un lenguaje de programación al lenguaje máquina del ordenador.

Un traductor es un metaprograma que tiene como entrada un programa (o parte de un programa) escrito en lenguaje simbólico, alejado de la máquina, denominado *programa fuente* (*código fuente*) y proporciona como salida otro programa semánticamente equivalente, escrito en un lenguaje comprensible por el hardware del ordenador denominado *programa objeto* (*código objeto*).

En este material se tratarán dos tipos de traductores, los compiladores y los intérpretes, que representan dos aproximaciones muy diferentes a la tarea de permitir el funcionamiento de los programas escritos en un determinado lenguaje de programación de alto nivel.

Compiladores

Un compilador traduce completamente un programa fuente y genera un programa objeto (semánticamente equivalente) escrito en lenguaje máquina.

Como parte importante de este proceso de traducción, el compilador informa al desarrollador de la presencia de errores en el programa fuente y pasa a la creación del programa objeto (en lenguaje máquina) sólo en caso de que no se hayan detectado errores (generalmente, se suele cancelar la compilación cuando se detectan errores).

El programa fuente suele estar contenido en un fichero y el programa objeto se puede almacenar como otro fichero en memoria masiva (disco duro, CD, etc.) para ser ejecutado posteriormente sin que sea necesario volver a traducirlo.

Para saber más

Podéis ver los dispositivos externos de memoria masiva en el apartado 1.4 del módulo "Ordenadores y sistemas operativos".

Las principales ventajas de los compiladores frente a los intérpretes es que sólo se ha de efectuar la traducción una vez. Una vez traducido un programa, su ejecución es independiente de su compilación. Así, los compiladores permiten realizar optimizaciones de código al leer y traducir todo el texto.

Intérpretes

Un programa intérprete permite que un programa fuente escrito en un lenguaje determinado se vaya traduciendo y ejecutando directamente, sentencia a sentencia, por el ordenador.

El intérprete revisa una sentencia fuente, la analiza, la interpreta y da lugar a su ejecución inmediata.

A diferencia del proceso de "compilación", en este caso no se crea ningún fichero o programa objeto almacenable en memoria masiva para posibles ejecuciones futuras.

Si se utiliza un intérprete para traducir un programa, cada vez que se ejecuta este último se vuelve a analizar (ya que no se genera un fichero objeto).

En cambio, con un compilador, aunque resulte más lenta, sólo se ha de realizar la traducción una vez.

Además, los traductores no permiten realizar optimizaciones del código (que elimina órdenes innecesarias compactando el código) más allá del contexto de cada sentencia del programa.

La principal ventaja de los intérpretes sobre los compiladores es que resulta más fácil localizar y corregir errores de los programas (depuración de programas), ya que la ejecución de un programa bajo un intérprete se puede interrumpir en cualquier momento para conocer los valores de las diferentes variables y la instrucción fuente que se acaba de ejecutar en aquel momento.

Por este motivo, los intérpretes resultan más pedagógicos para aprender a programar, ya que el alumno puede detectar y corregir más fácilmente sus errores. Son famosos los intérpretes del lenguaje de programación Basic, un lenguaje muy utilizado para enseñar a programar.

2.1.3. Principales lenguajes de desarrollo

En este apartado se enumeran, brevemente, algunos de los principales lenguajes de programación.

El objetivo es tener una visión de diferentes tipos de lenguajes de desarrollo y su principal aplicación.

Se trata de conocer los principales conceptos de su utilidad y principal funcionalidad, pero no de saber la sintaxis y la manera de trabajar.

Dado que los lenguajes de alto nivel son los más utilizados, tal como se ha comentado en los apartados anteriores, la clasificación y presentación se centrará en estos lenguajes.

Los lenguajes de alto nivel se pueden clasificar en dos divisiones, los lenguajes de propósito general, que se pueden utilizar en cualquier tipo de aplicaciones (como, por ejemplo, el lenguaje C o el Pascal) y los lenguajes de propósito especial.

Una primera clasificación de los lenguajes de desarrollo se puede establecer entre los lenguaje de propósito general y los de propósito especial.

Aun así, al ser tan general esta clasificación, también se podrían intentar subclasificar desde el punto de vista del campo de aplicación al que pertenece el lenguaje:

- **Aplicaciones científicas**: predominan las operaciones numéricas o matriciales propias de algoritmos matemáticos. Un lenguaje adecuado para este tipo de aplicaciones es el Fortran, por ejemplo.

Para saber más al respecto

Para más información sobre lenguaje de base de datos SQL podéis consultar el módulo 2, "Bases de datos".

Fortran

El Fortran se diseñó para ser utilizado en aplicaciones científicas y técnicas (matemáticas, ciencia e ingeniería). Se caracteriza por la potencia en sus cálculos matemáticos, pero está limitado en todo lo que hace referencia al tratamiento de datos no numéricos, por lo que no resulta adecuado para aplicaciones de gestión, gestión de ficheros y el tratamiento de caracteres y edición de informes.

A pesar de haberse creado en 1953, de la mano de John Backups, un empleado de IBM, sigue siendo un lenguaje común en aplicaciones de investigación, ingeniería y educación.

Fortran significa ***for**mula **tra**nslator* ('traductor de fórmulas') y se considera el primer lenguaje de alto nivel; la primera versión apareció en 1957.

- **Aplicaciones de procesamiento de datos**: en estas aplicaciones son frecuentes las operaciones de creación, mantenimiento y consulta sobre ficheros y bases de datos. Dentro de este campo, se encontrarían las aplicaciones de gestión empresarial, como los programas de nóminas, contabilidad, facturación, control de inventario, etc. Algunos de los lenguajes aptos para este tipo de aplicaciones son el Cobol y el lenguaje de bases de datos SQL.

Cobol

El Cobol es el lenguaje más utilizado en aplicaciones de gestión. Lo creó en 1960 un comité patrocinado por el Departamento de Defensa de Estados Unidos con la finalidad de disponer de un lenguaje universal para aplicaciones comerciales, y hoy día existen millones de líneas de código escritas en este lenguaje.

El nombre Cobol proviene de la frase *common business oriented language* ('lenguaje general para los negocios') y a lo largo de su existencia ha sufrido distintas actualizaciones.

Las características más interesantes de este lenguaje son que se parece al lenguaje natural (se utiliza mucho el inglés sencillo), es autodocumentado y ofrece grandes facilidades en la gestión de ficheros y también en la edición de informes escritos.

En cambio, sus rígidas reglas de formato de escritura, la necesidad de escribir todos los elementos al máximo detalle, la extensión excesiva de sus sentencias (a veces resulta demasiado enrevesado y reiterativo) y la inexistencia de funciones matemáticas son sus inconvenientes principales.

- **Aplicaciones de tratamiento de textos**: estas aplicaciones se encuentran asociadas al manejo de textos en lenguaje natural. El lenguaje C sería adecuado para este tipo de aplicaciones.

Lenguaje C

El lenguaje C lo creó en 1972 Dennis Ritchie (que, junto con Ken Thompson, había diseñado anteriormente el sistema operativo Unix), con la intención de conseguir un lenguaje idóneo para la programación de sistemas que fueran independientes de la máquina para utilizarlos en la implementación del sistema operativo Unix.

Desde entonces, tanto Unix como C han tenido un enorme desarrollo y proliferación, hasta convertirse en un estándar industrial para el desarrollo del software.

El C es un lenguaje moderno de propósito general que combina las características de un lenguaje de alto nivel (programación estructurada, tipo y estructuras de datos, recursividad, etc.) con una serie de características más propias de lenguajes de bajo nivel.

Esta cualidad del C posibilita que el programador utilice la programación estructurada para resolver tareas de bajo nivel, obteniendo un código ejecutable veloz y eficiente.

Por ello, mucha gente lo considera como un lenguaje de nivel medio.

- **Aplicaciones en inteligencia artificial**: destacan las aplicaciones en sistemas expertos, juegos, visión artificial y robótica; los lenguajes más populares dentro de este ámbito son el Lisp y el Prolog.

Lisp

El Lisp fue diseñado en 1959 por John McCarthy en el MIT (Instituto Tecnológico de Massachusetts) para utilizarlo en el ámbito de la inteligencia artificial. Este lenguaje toma su nombre del inglés ***list p****rocessing* ('procesamiento de listas').

Lisp está pensado para resolver problemas de manipulación de símbolos (que son de gran interés en inteligencia artificial). Los símbolos son elementos básicos de este lenguaje y representan objetos arbitrarios del dominio de interés que se esté tratando.

Resulta un lenguaje funcional, ya que todo programa (o subprograma) en Lisp se puede ver como una función de alto nivel que se aplica sobre otras funciones de más bajo nivel para obtener determinados resultados. Para realizar operaciones elementales se pueden utilizar funciones de una biblioteca.

Este lenguaje no se parece nada a otros lenguajes de programación. Aun así, resulta un lenguaje fácil de entender y es el más común dentro de las aplicaciones en inteligencia artificial.

Uno de sus principales problemas era que no se podía ejecutar de manera eficiente en muchos ordenadores. En la actualidad, existen versiones estándar de Lisp, Common Lisp, DG Common Lisp y Lisp portable estándar.

Prolog

El Prolog (***pro****gramming* ***logi****c*) se desarrolló a partir del trabajo realizado en la década de los años setenta, principalmente en universidades europeas, y ha sido el lenguaje más utilizado en el ámbito de la inteligencia artificial en Europa.

Se basa en la lógica y ha sido utilizado para desarrollar un gran número de aplicaciones en bases de datos e inteligencia artificial.

El Prolog permite que el programador exprese una serie de tareas basadas en la descripción de los objetos que intervienen (hechos y reglas) y las relaciones lógicas que existen entre ellos (predicados), en lugar de hacerlo mediante un algoritmo.

Lleva incorporada la programación de operaciones y todo el esfuerzo de programación consiste en especificar adecuadamente los hechos y las reglas para después establecer preguntas que se podrán inferir de manera automática.

El Prolog permite desarrollar sistemas expertos a personas sin mucha idea de programación, ya que no requiere programar ningún algoritmo.

Vista esta facilidad de uso, una importante aplicación del Prolog es la educación, donde se puede utilizar para enseñar lógica, técnicas de resolución de problemas y se suele emplear en un gran número de bases de datos educativas.

- **Aplicaciones de programación de sistemas**: en este campo se incluirían la programación de software de interfaz entre el usuario y el hardware, como son los módulos de un sistema operativo y los traductores. Tradicionalmente, tal como se ha mencionado con anterioridad, para estas aplicaciones se utilizaba el lenguaje ensamblador. En la actualidad se muestran muy adecuados los lenguajes como el Ada, el Modula-2 y el C, por ejemplo.

Ada

El lenguaje Ada nació con el objetivo de obtener un único lenguaje para todo tipo de aplicaciones (un auténtico lenguaje de propósito general).

Lo encargó el Departamento de Defensa de Estados Unidos y su estandarización fue publicada en 1983. El nombre de Ada proviene de Augusta Ada Byron, condesa de Lovelace, considerada la primera programadora de la historia.

Entre las características del lenguaje se incluye la compilación separada, la programación concurrente, la programación estructurada, la buena mantenibilidad, características de tiempo real, etc.

Sin embargo, el principal inconveniente de este lenguaje es su gran extensión, que puede complicar su uso.

Modula-2

El mismo Nicklaus Wirth, creador del lenguaje Pascal, dirigió, a finales de los años setenta, el desarrollo del lenguaje Modula-2 (denominado en un principio simplemente Modula), con la intención de incluir las necesidades de la programación de sistemas y responder a las críticas recibidas con respecto a las carencias del lenguaje Pascal.

Además de las principales características del Pascal, este lenguaje incorpora funcionalidades como la posibilidad de compilación separada, la creación de bibliotecas, la programación concurrente, la mejor gestión de cadenas de caracteres, los procedimientos de entrada/salida y de gestión de la memoria y, además, aporta grandes facilidades para la programación de sistemas.

Este lenguaje también posee cualidades didácticas, por lo que ha sido sobradamente aceptado dentro de la comunidad universitaria como herramienta idónea para enseñar la programación.

Otra clasificación según el tipo de aplicación del lenguaje permitiría clasificarlos en aplicaciones científicas, de procesamiento de datos, de tratamiento de textos, de inteligencia artificial y de programación de sistemas.

Otra manera de clasificar los lenguajes de alto nivel puede ser según el estilo de programación que fomentan, es decir, la filosofía de construcción de programas:

- Lenguajes imperativos o procedimentales: establecen cómo se debe ejecutar una tarea, dividiéndola en partes que especifican cómo realizar cada una de las subtareas asociadas. Estos lenguajes se fundamentan en el uso de variables para almacenar valores y el uso de instrucciones que indican las operaciones que debe realizar sobre los datos almacenados. La mayoría de los lenguajes de alto nivel son de este tipo: Fortran, Basic, Pascal, Ada, Modula-2, Algol, RPG, PL/1, Simula, Smalltalk y Eiffel.

Basic

El lenguaje Basic se diseñó en 1965 con el objetivo de que lo utilizaran los principiantes en su introducción al mundo de la programación.

Resulta un lenguaje fácil de aprender, como indica su nombre: ***b**eginner's **a**ll-purpose **s**ymbolic **i**nstruction **c**ode* ('código simbólico de propósito general para principiantes'); al principio se enfocó a enseñar a estudiantes a los que se pretendía introducir en el mundo de la programación y se convirtió en el lenguaje educativo más popular del mundo.

Las principales aportaciones de Basic son que es un lenguaje interpretado (traducido por un programa intérprete) y que es de uso interactivo.

Con los años, este lenguaje ha evolucionado hacia versiones preparadas para desarrollar aplicaciones por Internet, de gestión y soportar características y métodos orientados a objetos, como el Visual Basic.

Pascal

El lenguaje Pascal lo desarrolló en 1970 el matemático suizo Nicklaus Wirth con el objetivo de proporcionar un lenguaje adecuado para la enseñanza de los conceptos y las técnicas de programación, y permitió el desarrollo de aplicaciones fiables y eficientes en los ordenadores disponibles en aquel momento.

El lenguaje recibe su nombre en honor del filósofo y matemático francés Blaise Pascal, que inventó la primera máquina de tipo mecánico para sumar y, con el tiempo, ha llegado a ser un lenguaje muy utilizado en todo tipo de aplicaciones, y se emplea mucho en las universidades para aplicaciones científicas y de ingeniería.

Pascal se ha diseñado para ilustrar conceptos clave en programación, como los de tipo de datos, programación estructurada (es un lenguaje muy estructurado) y diseño descendente. El Pascal trata de proporcionar un mecanismo para implementar cada uno de los conceptos de programación.

Este lenguaje se ha convertido en el predecesor de otros lenguajes más modernos, como el Modula-2 y el Ada.

Algol

El Algol (***algo****rithmic* ***l****anguage* o ***alg****ebraic* o ***o****rented* ***l****anguage*) es un lenguaje algorítmico u orientado al álgebra, importante por haber introducido por primera vez conceptos que se han revelado básicos en la programación, principalmente la idea de estructuración. Se llegó a utilizar como un lenguaje internacionalmente adoptado para la descripción de algoritmos matemáticos. Se desarrolló en la década de los sesenta.

RPG

El RPG (*report program generator*) es en realidad un generador de programas que lee unas especificaciones y adapta los módulos de programa para la realización de las entradas y salidas indicadas en las instrucciones.

Apareció con los pequeños ordenadores para gestión como el Sistema/3 de IBM a finales de los años sesenta. Modificaciones posteriores (las tarjetas de instrucciones tipos C y las versiones RPG III y RPG 400) lo configuran con la posibilidad de desarrollar algoritmos específicos, mediante el uso de variables binarias conocidas como *interruptores*, y también de gestionar bases de datos.

PL/1

El PL/1 (*program language 1*) es un lenguaje diseñado para que se utilice tanto en los problemas de tipo científico, como en los de gestión. Su diseño se basa mucho en otros lenguajes como el Fortran, el Cobol y el Algol.

Del Algol adopta la estructura de bloques y las instrucciones de control de secuencias estructuradas; del Cobol, la riqueza en la gestión de ficheros y la declaración de datos de tipo PICTURE, y del Fortran, la forma concisa y simple de las instrucciones y el mecanismo de transmisión de parámetros, entre otros elementos.

Simula

Es el antecesor de los lenguajes orientados a objetos. Construido en torno a ideas ya presentes en el Algol, en 1962, el Simula incluye, además, los conceptos de encapsulación y herencia, y fue pionero a la hora de utilizar los conceptos de clase, objetos, mensajes y tipos abstractos de datos, tan corrientes, tiempo después, en todos los lenguajes orientados a objetos.

Uno de los aspectos más importantes de Simula, concebido como un lenguaje que describe sistemas y elabora simulaciones de estos sistemas por ordenador, es que introdujo la idea de construir simulaciones de sistemas complejos procurando reflejar el vocabulario del sistema real y del dominio del problema.

Smalltalk

Se puede considerar el primer lenguaje orientado a objetos realmente "puro", en el sentido de que es el primero que se plantea como un lenguaje de nuevo acuñamiento para adoptar íntegramente la nueva orientación a objetos. En el Smalltalk todo se trata como un objeto, desde los números enteros hasta las clases.

Concebido como un intérprete, se adelantó a su tiempo en la utilización de una interfaz gráfica y el uso interactivo. Su influencia ha resultado decisiva en el resto de lenguajes orientados a objetos.

Eiffel

Eiffel se publicó en 1986 y se presentó como un lenguaje orientado a objetos que se traduce primero a otro lenguaje de programación, lo cual, como en el caso de C++, favorece la portabilidad.

El lenguaje Eiffel se considera, en el mundo académico, como el lenguaje orientado a objetos mejor diseñado y completo, aunque no siempre resulte el más eficiente.

En realidad, es un entorno de desarrollo con herramientas auxiliares, como sistemas gráficos de exploración de estructuras, depuradores, apoyo de persistencia, gestión de configuraciones, etc.

- Lenguajes declarativos: en este caso, el proceso por el cual se ejecuta el programa no aparece de manera explícita en el programa. El programador no necesita indicar el proceso detallado de cómo realizar la tarea. En estos lenguajes, los programas se construyen mediante descripciones de funciones (lenguajes funcionales, como Lisp) o expresiones lógicas que indican las relaciones entre determinadas estructuras de datos (lenguajes de programación lógica, como Prolog).

- Lenguajes orientados a objetos: el diseño de los programas se centra más en los datos y su estructura. Los programas consisten en descripciones de unidades denominadas objetos que encapsulan los datos y las operaciones que actúan sobre éstos. El lenguaje más utilizado dentro de este tipo es el C++ .

- Lenguajes orientados al problema: este tipo de lenguajes están diseñados para problemas específicos, principalmente de gestión. En estos lenguajes, los programas están formados por sentencias que ordenan qué se quiere realizar. Generalmente, estos lenguajes suelen ser generadores de aplicaciones que permiten automatizar tanto como pueden la tarea de desarrollo del software de aplicación de gestión.

Finalmente, si se tiene en cuenta la manera de programar, utilizando estos lenguajes se pueden categorizar entre los procedimentales, los declarativos, los orientados a objetos y los orientados al problema.

Java

Es un lenguaje de programación orientado a objetos creado por Sun Microsystems, a principios de los años noventa. Es similar al lenguaje C++ pero no tiene elementos de bajo nivel, por lo que resulta conceptualmente menos abstracto.

Es un lenguaje interpretado; el lenguaje de alto nivel es compilado en bytcodes, los cuales son transversales a cualquier arquitectura; por lo tanto, el programa será exportable a todas las arquitecturas de microprocesadores. Para que funcionen, sólo hace falta que la plataforma tenga una máquina virtual Java, es decir, el intérprete de los bytecodes adecuado a cada plataforma.

Javascript

Es un lenguaje orientado a objetos de forma amplia utilizado para dotar de dinamismo a las paginas webs escritas en código HTML (*hiper text marked language*). De sintaxis muy similar a Java, pero de complejidad mucho menor.

Fue inventado por Brendan Ecih mientras formaba parte de Netscape Comunications. Actualmente, todos los navegadores web soportan este lenguaje inevitablemente, ya que un porcentaje elevadísimo de las páginas webs lo usan.

Lenguajes propietarios

Algunas grandes aplicaciones se pueden personalizar (customizar) según la empresa que las compre para adaptarlas a su negocio y a su particular forma de operar, a la hora de adaptar la aplicación al negocio y después personalizarla; según sus flujos operativos particulares, hace falta algún lenguaje que haga este paso. Estos lenguajes no sirven para crear aplicaciones desde cero, sino para personalizar y adaptar aplicaciones de carácter genérico como SAP (ERP) o Siebel (CRM).

ERP

Enterprise resource planning (planificación de los recursos de la empresa).

CRM

Customer relationship mangement (gestión de la relación con el cliente).

2.2. Herramientas CASE

El término CASE (*computer aided software engineering*) se popularizó en la década de los ochenta para designar una tecnología que no era nueva: las ayudas automatizadas de las metodologías de desarrollo de software, y representaba la evolución de las primeras ayudas (editores, compiladores, etc.) que formaron los entornos de desarrollo de software de los años setenta.

2.2.1. Principales conceptos e historia

CASE, la ingeniería de software asistida por ordenador, incluye el conjunto de herramientas y métodos asociados que pueden servir de ayuda en el proceso de construcción del software a lo largo de su ciclo de vida.

La novedad del CASE se hizo evidente cuando se incorporaron a las herramientas de ayuda al diseño informático nuevas funcionalidades gráficas con las cuales se pudo representar, modificar y mantener actualizados los diagramas gráficos que se han convertido en clásicos y se utilizan en la mayoría de las metodologías de desarrollo de software.

Una herramienta CASE permite ayudar al desarrollador y al equipo de proyectos a construir el programa dándoles herramientas para que puedan generar documentación, código, diagramas, etc.

Las verdaderas herramientas CASE surgieron cuando los diagramas se incorporaron a una base de datos global de diseño (diccionario de datos o repositorio), que también mantenía detalles de los elementos de datos y de la lógica de los procesos.

Este diccionario de datos es, en realidad, una base de datos que incorpora un modelo en el que se incluyen restricciones de integridad de los datos.

Ejemplos de herramientas CASE son los sistemas Excelerator, System Architect, EasyCASE, IEF (*information enineering facility*), IEW (*information engineering workbench*), Teamwork, VAW (*visible analyst workbench*), así como todos los aparecidos más adelante que aprovechan al máximo todas las prestaciones de la interfaz gráfica.

Dos herramientas o entornos muy conocidos hoy día de herramientas CASE son el programa Microsoft Visio y el paquete Rational Rose, que incorpora aplicaciones para generar documentación, código, probar los programas, etc.

Si bien existen diferentes subdivisiones, una clasificación de las herramientas CASE se basa en CASE alto (*upper CASE*), medio (*middle CASE*) y bajo (*lower CASE*) de acuerdo con el nivel de abstracción y generalidad.

Otra distinción posible se establece entre herramientas (*toolkits*), supuestamente aplicables a una única tarea del desarrollo del software, y talleres de trabajo (*workbench*), que se presentan como una colección de herramientas integradas que ayudan a todo el proceso de desarrollo de software: análisis, diseño e implementación.

Existe gran infinidad de herramientas CASE en el mercado y se clasifican por CASE alto, medio y bajo o también por herramientas y talleres de trabajo.

2.2.2. Aplicación del CASE y Herramientas

La mayoría de las herramientas CASE se aplican a las fases de análisis y diseño, pero algunos sistemas más completos incorporan generadores de lenguajes e incluyen la posibilidad de generar el código fuente de la aplicación de una manera automática.

De esta manera, ayudan tanto al proceso de construcción del software, como también a la elaboración y al mantenimiento de la documentación, con lo que simplifican buena parte del trabajo de programar.

A veces, las herramientas CASE son simplemente un generador que utiliza las especificaciones procedentes de otra herramienta CASE, como puede suceder con la familia de generadores del APS Development Center, que puede generar el Cobol a partir, por ejemplo, de las especificaciones que le proporciona Excelerator.

Con respecto al análisis y diseño de sistemas, predominan las herramientas que implementan los diagramas de flujo de datos (DFD, *data flow diagrams*) del análisis y diseño estructurado de Constantine, Yourdon y De Marco y también en la variante de Gane/Searson.

En cuanto al diseño de datos, se utiliza a menudo el diagrama de entidad-relación de Chen (ERS, *entity- relationship diagrams*) o la versión más clásica de los diagramas de Bachman para convertirlos finalmente en su equivalente en el modelo de bases de datos relacionales.

También, para nuevas metodologías como MERISE, se implementan los diagramas de la historia de vida de una entidad (ELH, *entity-life-history-diagrams*).

Otras técnicas de diseño muy populares y utilizadas por las herramientas CASE son los diagramas de descomposición y jerarquía de funciones (HD, *hierarhical diagram*), los diagramas de estructura de módulos de la programación modular y el diseño estructurado (SC, *structured chart*), los diagramas de estructura de datos de la metodología Jackson (DSD, *data structure diagrams*) o los grafos de diseño de diálogos (IDS, *invocation dialog structure*) y los diseños de pantallas y listados entre otras modelizaciones gráficas muy utilizadas en la ingeniería del software.

Objetivo del apartado

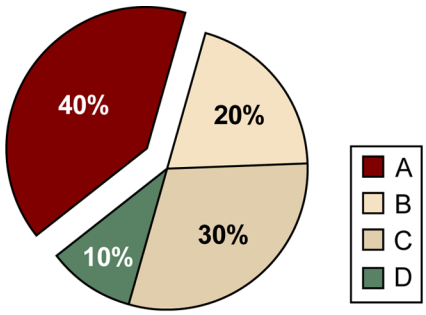
El objetivo de este apartado no es profundizar en técnicas y herramientas de diseño y desarrollo de aplicaciones. Aun así, se enumeran todas estas técnicas para que resulten familiares al estudiante.

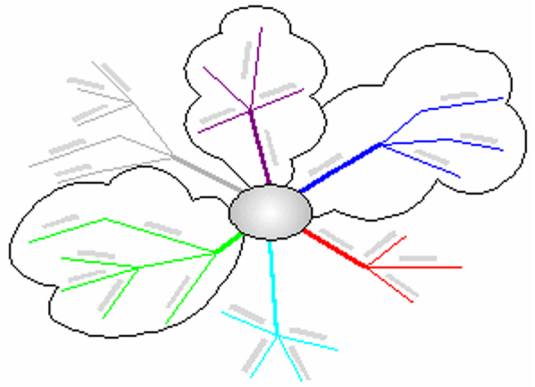
Las herramientas CASE pueden tener diferentes aplicaciones. Desde el análisis y diseño hasta la generación del programa y su documentación.

Algunos de los modelos que permiten generar estas herramientas son los siguientes:

- Diagramas genéricos. Utilidades para crear diseños y esquemas para planificación, comunicación y generación de documentación de apoyo.

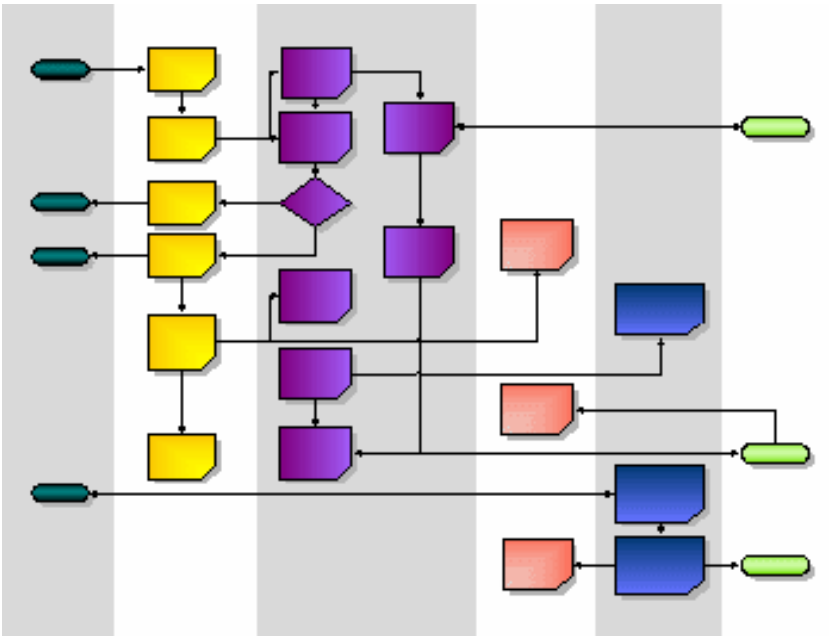
Diagramas genéricos





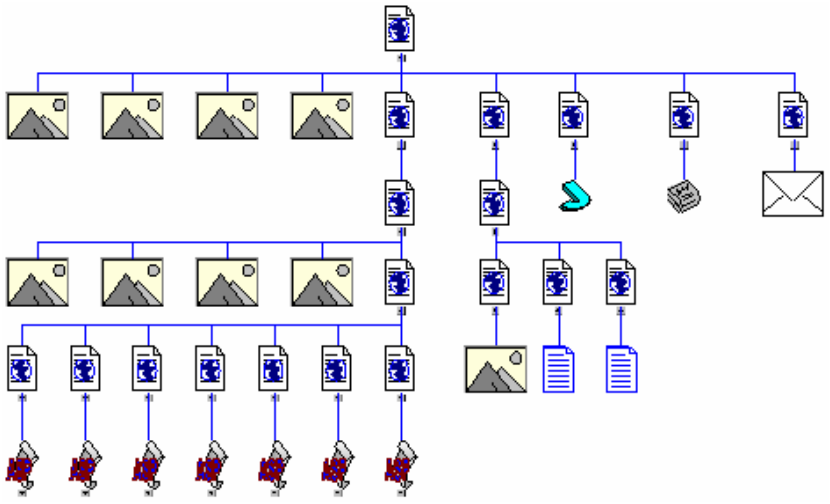
- Diagramas de auditoría. Utilizados para documentar y analizar procesos.

Diagramas de auditoría



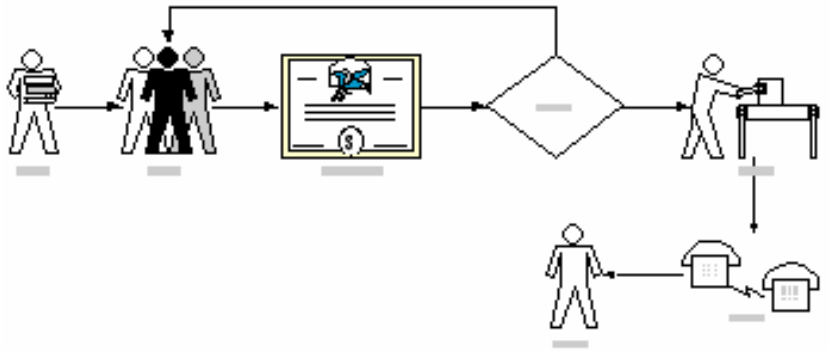
- Diagramas "conceptuales de sitios web". Utilizados en el diseño y desarrollo de webs.

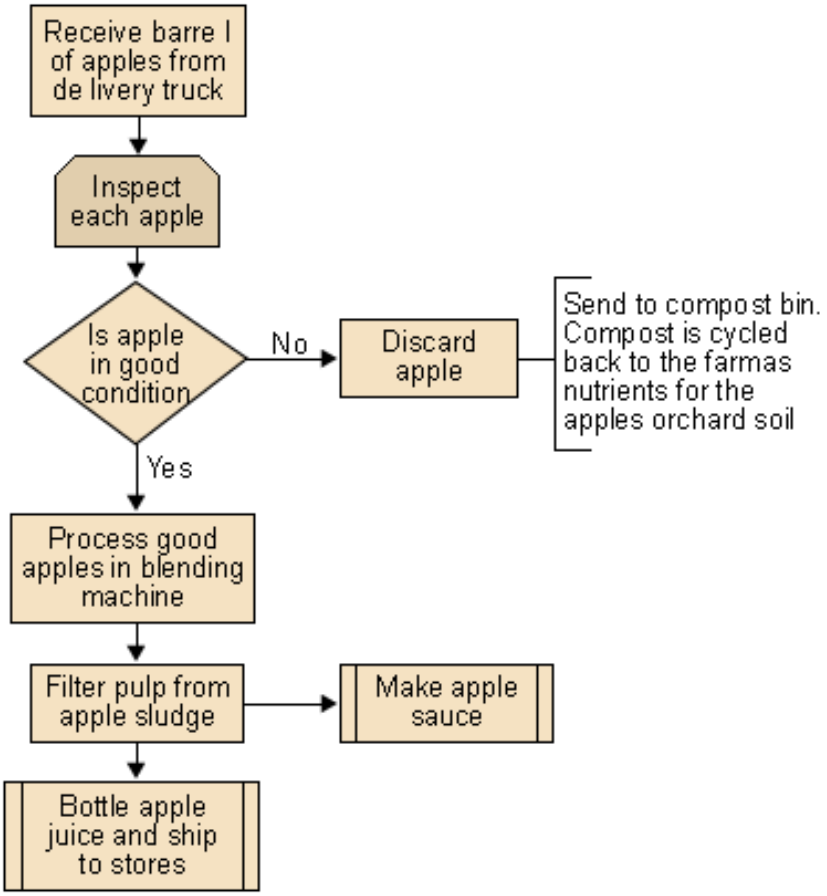
Diagramas "conceptuales de sitios web"



- Diagramas de procesos y flujos de datos. Permiten diseñar y reflejar los procesos de negocio y los flujos de datos y documentos de los diferentes departamentos de una compañía y su interacción.

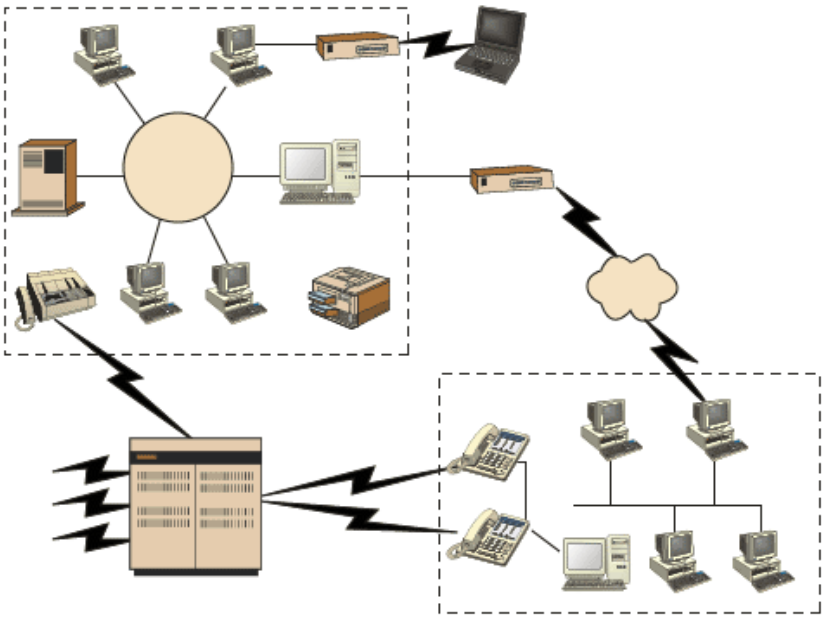
Diagramas de procesos y flujos de datos





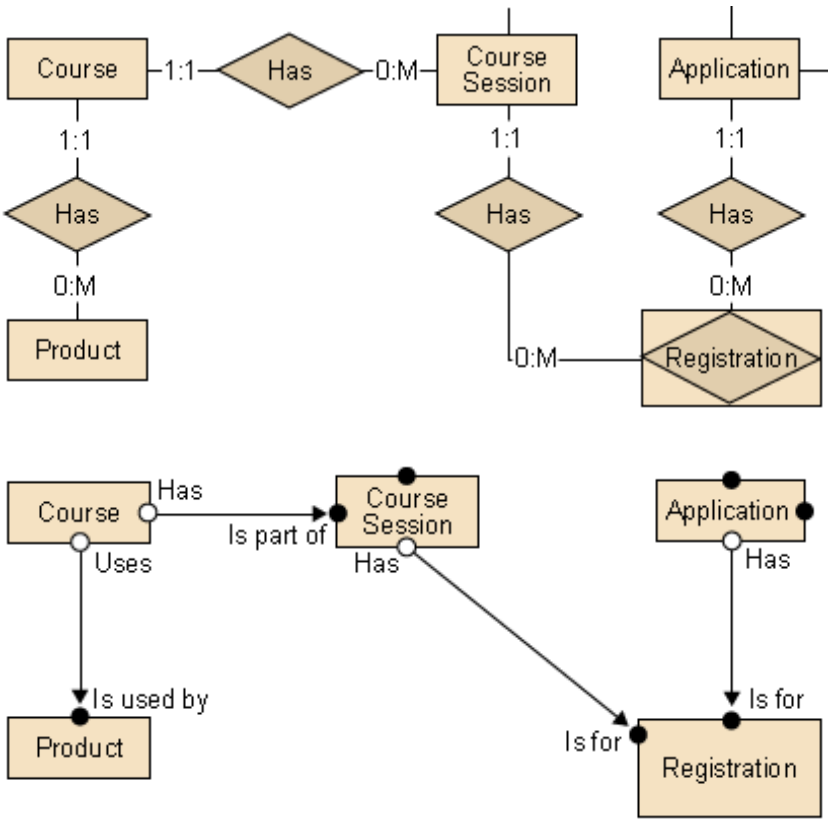
- Diagramas de redes. Permiten crear y diseñar el diagrama tanto lógico como físico de una red de ordenadores y de sus dispositivos.

Diagramas de redes



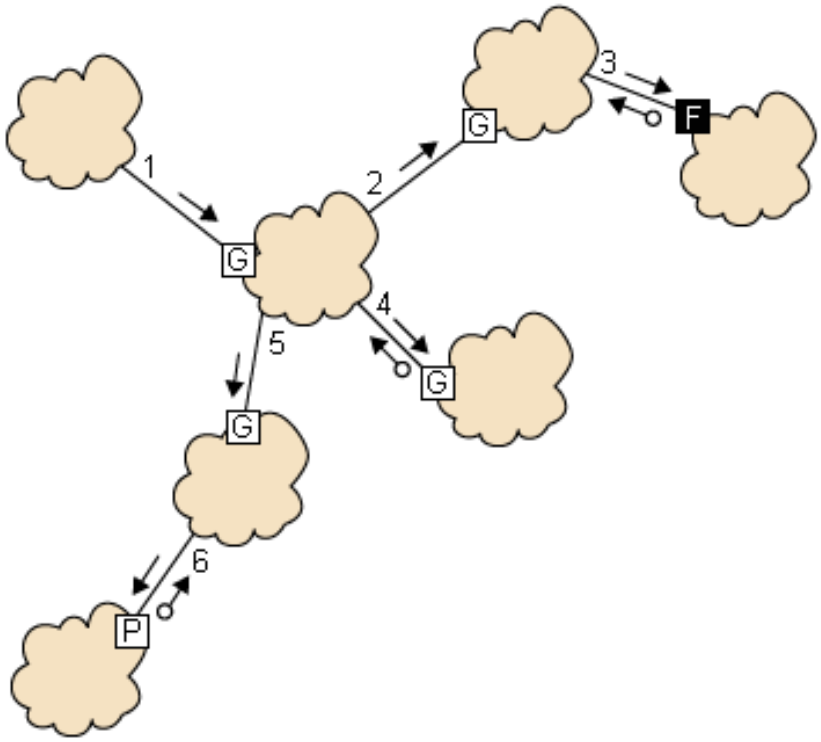
- Diagramas de modelos conceptuales de BD.

Diagramas de modelos conceptuales de BD



- Diagramas de modelos "orientados a objetos". A partir de diferentes notaciones (Booch OOD, los casos de uso de Jacobson, la notación ERD Martin, el lenguaje de modelización ROOM o la metodología OMT de Rumbaugh), este tipo de diagramas permiten diseñar aplicaciones orientadas a objetos. La mayoría de las herramientas CASE también poseen funciones para generar código de programación a partir de estos diseños.

Diagramas de modelos "orientados a objetos"



2.3. Introducción a la ingeniería del software

El objetivo de este material no consiste en cubrir y tratar la ingeniería del software, sino en ofrecer una breve introducción y visión de conjunto de sus principales conceptos.

La ingeniería del software nació de la necesidad de estandarizar el proceso de producción del software y minimizó el riesgo de errores en su desarrollo y mantenimiento.

A continuación, se describen las principales fases de la producción del software, la importancia del mantenimiento de los programas y los principales elementos de la ingeniería del software.

2.3.1. La producción de software

Tal como se ha mencionado en los apartados anteriores, el objetivo y el uso de los lenguajes de programación es la construcción de software, es decir, aplicaciones que se ejecutarán en un sistema informático (hardware, red, etc.) con el fin de llevar a cabo un conjunto de funcionalidades y operaciones.

El proceso de producción del software tiene una estructura general donde se podrían distinguir tres fases genéricas: la definición, el desarrollo y el mantenimiento.

Las tareas que se realizan durante cada una de estas fases de producción son las siguientes:

- **Definición**: esta fase se focaliza en el qué, es decir, "qué cosa" se ha de desarrollar, y en ella se define qué información se debe procesar, qué función y rendimiento se desea del futuro programa, qué restricciones se han de imponer y qué criterios de validación son necesarios para que el futuro sistema desarrollado se pueda considerar correcto.

Toma de requerimientos

Dentro de la definición, está la toma de requerimientos al solicitante del software. A menudo, quien pide el nuevo software conoce perfectamente el problema que tiene y que quiere resolver con el nuevo software, pero no tiene claro la solución que resuelve su problema. Para evitar que la fase de definición se convierta en algo imposible de abarcar, hace falta que el ingeniero de software conozca muy bien la tecnología con la que quiere implementar la solución y que hable con todos los implicados en el problema, desde los usuarios a los niveles jerárquicos más elevados. Una vez se han recogido todos los datos y deseos de todos los implicados, cabe presentar una propuesta que contente a todas las partes, solucione el problema y sea implementable técnicamente dentro del tiempo pactado y dentro del presupuesto que se tenga. Prometer soluciones poco realistas hará fracasar el proyecto.

- **Desarrollo**: la segunda fase se centra en el cómo. Así, durante esta fase es necesario decidir cómo transformar aquello definido anteriormente en un producto software, cómo se han de diseñar las estructuras de datos y la arquitectura de software, cómo se implementan los diferentes detalles procedimentales, cómo se ha de trasladar el diseño a un lenguaje de programación y cómo se debe realizar la prueba de validación una vez generado el programa.

La tecnología

Acertar la tecnología con la que se quiere desarrollar el software puede ayudar mucho al éxito del mismo. Un buen desarrollo siempre es más ágil si detrás hay un buen análisis y las herramientas de desarrollo adecuadas, tanto de software como de hardware. Disfrutar de los perfiles oportunos es también de vital importancia para esta fase.

- **Mantenimiento**: finalmente, y una vez que el programa se ha desarrollado, se entra en la fase de realizar y llevar a cabo cambios asociados con la corrección de errores, la adaptación a nuevos entornos y el aumento de la funcionalidad.

Tipo de mantenimiento

Hay dos tipos básicos de mantenimiento: el correctivo y el evolutivo. El mantenimiento correctivo tiene una gran actividad en los primeros meses de existencia del software, pero poco a poco tiende a ir desapareciendo a medida que se van abarcando todos los errores. El mantenimiento evolutivo, en cambio, tiene una gran actividad durante todo el ciclo de vida del software, ya que consiste en corregir, pero también en evolucionar y mejorar, el software a medida que surgen nuevas necesidades o ideas. A menudo, algunos cambios del mantenimiento evolutivo necesitan ser tratados como un pequeño proyecto de software, con su toma de requerimientos, análisis, presupuesto, etc.

Las tres principales fases del desarrollo del software son la definición, el desarrollo y el mantenimiento posterior.

2.3.2. El mantenimiento del software

El software no se deteriora, como sí que le sucede al hardware.

El hardware presenta relativamente muchos errores al principio de su vida (construcción), a menudo atribuibles a defectos de diseño o de fabricación o de especificaciones.

Cuando se corrigen los defectos, disminuyen los errores hasta el nivel más bajo, donde quedan estacionarios durante un determinado período de tiempo, hasta que llega un momento de su vida en el que empieza a mostrar de nuevo errores como consecuencia del paso del tiempo y de otros daños externos que deterioran los componentes.

Aun así, la realidad es que el software queda obsoleto antes de que se deteriore.

Una de las razones de esta obsolescencia se debe a la rapidez con la que evoluciona el software, que provoca que cada vez se exijan más recursos de hardware.

Otra razón es la evolución del entorno empresarial, científico, docente, etc., que hace que los flujos de trabajo vayan variando y que el software ya no responda a todas las nuevas necesidades surgidas.

Con frecuencia, al solventar los errores iniciales, se incluyen modificaciones que hacen que aparezcan o salgan a la luz nuevos errores. Para evitarlo, hay que enfatizar la rigurosidad de los análisis y las fases de test.

En general, el mantenimiento del software tiene una complejidad considerablemente mayor que el mantenimiento del hardware.

Un error en el software se debe a incoherencias en las especificaciones, a un error en el diseño o a un desarrollo mal resuelto. No existen piezas de recambio, como en el caso del hardware; eso obliga a que toda solución se tenga que resolver con más desarrollo, algo también sometido a la posibilidad de contener errores. Así, en el software no existen piezas de recambio, pero un buen diseño para ayudar a simularlas. Todas las tareas del software se pueden diseñar e implementar como módulos individuales que interactúan entre ellas, de manera que si una parte queda obsoleta por la tecnología se puede rehacer sólo el módulo implicado. Tener el código modularizado también facilita el mantenimiento y sus reos en diferentes proyectos.

El software no se deteriora como el hardware. Aun así, un error en el diseño y desarrollo de un programa obliga a su reprogramación, ya que no hay piezas de recambio.

2.3.3. Principales elementos de la ingeniería del software

Vista la importancia de garantizar el menor número de errores en la producción del software y, al mismo tiempo, la necesidad de estandarizar el desarrollo de las aplicaciones, apareció la ingeniería del software.

La ingeniería del software engloba un conjunto de tres elementos clave (métodos, herramientas y procedimientos) que facilitan que el gestor controle el proceso de desarrollo del software y que suministran las bases para construir software de alta calidad de una manera productiva.

- Los métodos de la ingeniería del software indican cómo construir técnicamente el software e incluyen un amplio espectro de tareas, desde la planificación y estimación de proyectos, hasta el análisis de los requerimientos, el diseño, la codificación, la prueba y el mantenimiento.

- Las herramientas de la ingeniería del software suministran un apoyo automático o semiautomático para los métodos. En la actualidad, existen herramientas para soportar cada uno de los métodos mencionados anteriormente. La ingeniería del software asistida por ordenador, conocida normalmente como CASE (*computer aided software engineering*) (podéis ver "Herramientas CASE"), es una herramienta que permite la automatización del proceso de producción de desarrollo del software. La idea básica de una herramienta CASE es proporcionar un conjunto de herramientas integradas que ahorren trabajo enlazando y automatizando todas las fases del ciclo de vida del software.

- Los procedimientos de la ingeniería del software son el nexo de unión entre los métodos y las herramientas, y facilitan un desarrollo racional y oportuno del software. Los procedimientos definen la secuencia en la que se aplican los métodos, las entregas (documentos, informes, códigos, etc.) que se requieren, los controles que ayudan a asegurar la calidad y coordinar los cambios, así como las guías que facilitan a los gestores de los software establecer su desarrollo.

La ingeniería del software incluye y define los métodos, las herramientas y los procedimientos para el desarrollo del software.

3. Gestión y planificación de proyectos informáticos

3.1. La gestión de proyectos

Si se quisiera definir qué es un proyecto, se encontrarían distintos conceptos y definiciones.

Según la Asociación Francesa de Normalización, un proyecto es "un proceso específico que permite estructurar metódica y progresivamente una realidad futura".

Otra definición sería considerar un proyecto como "un conjunto de recursos humanos y materiales necesarios para conseguir un objetivo delimitado por unos costes, unos plazos y unos resultados".

Y, finalmente, atendiendo al Project Management Institute (PMI), un proyecto se define como "un conjunto de esfuerzos, temporales, dedicados a crear un servicio o producto único".

En todo caso, independientemente de la definición, se puede concluir que un proyecto presenta las características siguientes:

- Una delimitación temporal.
- Unos objetivos precisos de plazos, costes y resultados que se deben obtener.
- No es una acción repetitiva, sino una acción creativa.
- Está compuesto por un conjunto de tareas complejas que requieren un trabajo de análisis, estudio y planificación.

3.1.1. Gestión de un proyecto

La gestión de un proyecto es la planificación, organización y dirección de las tareas o actividades y de los recursos necesarios para conseguir un objetivo definido, limitado temporalmente y con un presupuesto económico fijado.

Así pues, la finalidad de la gestión de proyectos es conseguir un objetivo específico en una fecha máxima y con un presupuesto determinado.

Ejemplos de proyectos pueden ser:

- La construcción de un local, supermercado, edificio, etc.
- El lanzamiento de un nuevo producto: plan de marketing, plan de ventas.
- El desarrollo de un producto nuevo: plan de I+D, plan de desarrollo.
- La organización de acontecimientos deportivos o culturales.
- El desarrollo de una aplicación informática.
- El desarrollo del proceso de fabricación de un vehículo nuevo.
- La gestión de una campaña electoral.
- La implantación de nuevos procesos organizativos en una empresa.
- La instalación de un sistema informático.

Un proyecto u objetivo puede ser tan "simple" como diseñar el prospecto de publicidad de un producto nuevo o tan "complejo" como la construcción de un centro comercial.

En cada caso, es necesario:

- dividir el proyecto en unas tareas o actividades,
- programarlas en fechas,
- asignar los recursos necesarios
- y, finalmente, llevar a cabo un seguimiento para evaluar el progreso del trabajo y detectar posibles desviaciones y poder corregirlas.

La gestión de proyectos ayuda a responder a preguntas. Para poder evaluar si el proyecto tiene desviaciones o no, una buena gestión de proyectos tiene que poder contestar a:

- ¿Cuánto tiempo exigirá este proyecto?
- Si una tarea se retrasa, ¿cuánto tiempo se retrasará el proyecto?
- ¿Se cuenta con suficientes recursos, humanos y materiales, con el fin de completar el proyecto, tal como hemos definido?
- ¿Cuáles son los costes asociados a los recursos para el proyecto?

- En caso de retraso, ¿qué puedo hacer para recuperar tiempo?

- Si se quisiera adelantar la finalización del proyecto, ¿qué recursos se deberían añadir y qué impacto económico supondría?

La gestión de un proyecto engloba la planificación, la organización y la dirección de las tareas o actividades y de los recursos necesarios para conseguir un objetivo definido, limitado temporalmente y con un presupuesto económico fijado.

3.1.2. Las fases de la gestión de un proyecto

Generalmente, la gestión de proyectos agrupa tres fases: la definición y la planificación, la dirección y el seguimiento, y la generación de informes.

- **Definición y planificación de un proyecto**. Constituye la parte más importante, ya que incluye la definición de las tareas y su duración, el análisis de las relaciones entre las tareas y la asignación de recursos a cada una, principalmente.
  Todas las fases posteriores del proyecto se llevarán a cabo según la información procedente de este diseño.

- **Dirección y seguimiento del proyecto**. Es la fase de seguimiento del proyecto una vez que ha empezado y acaba con el proyecto.
  Incluye el seguimiento de las tareas definidas, la asignación prevista de los recursos y los reajustes necesarios para reflejar los cambios que se producen mientras avanza el proyecto.

- **Generación de informes sobre el proyecto**. Una vez acabado el proyecto, en esta fase se producen los diferentes informes finales.
  Aun así, durante la fase de dirección también se generan informes de seguimiento y durante la fase de creación se elaboran los informes de planificación.

En función del tipo de proyecto u objetivo que se ha de alcanzar, estas fases se subdividirán en más o menos subetapas, pudiendo hablar, incluso, de modelos de proyecto según su tipología: construcción de un edificio, levantamiento de un puente, elaboración de una campaña de publicidad, o la implantación de una herramienta CRM (*customer relationship management*) o el diseño e implantación de una web, en el ámbito de los proyectos informáticos.

3.2. Planificación y seguimiento

La planificación de un proyecto, tal como se ha visto en el apartado "Gestión de un proyecto", constituye la primera fase de todo proyecto y consiste en definir qué tareas serán necesarias para conseguir el hito marcado, qué recursos, materiales y humanos, será necesario gestionar, cuánto tiempo se necesitará para llevarlo a cabo y, finalmente, qué costes asociados tiene.

Una vez se ha iniciado el proyecto, se debe realizar un seguimiento con el fin de evaluar la progresión de las tareas, comprobar la realización de éstas, identificar posibles retrasos y gestionar las incidencias que surjan.

3.2.1. Las tareas de un proyecto

Un proyecto está constituido por un conjunto de tareas (o actividades).

Una tarea es necesaria para el proyecto si su ejecución contribuye a finalizarlo y, por otra parte, si su no ejecución conlleva una dificultad en la consecución de los objetivos.

Una tarea se determina por las características siguientes:

- nombre,
- duración estimada,
- su interrelación con otras tareas del proyecto,
- los recursos que requiere para ser realizada,
- una descripción sobre qué consiste.

Un tipo de tarea particular son los hitos, que corresponden a tareas de duración nula y que sirven para indicar puntos de control del proyecto.

Hitos

Ejemplos de hitos son el inicio del proyecto o la consecución parcial de objetivos.

Los hitos también pueden corresponder a tareas externas al proyecto como, por ejemplo:

- La concesión de una licencia de obras por parte del ayuntamiento.
- La aprobación de un crédito bancario para llevar a cabo la inversión necesaria del proyecto.
- Etc.

En estos casos, dado que la tarea no es controlable por el equipo de trabajo, no se considera ni un tiempo ni un esfuerzo concreto. Aun así, en estos casos, aunque la responsabilidad de la tarea es externa al proyecto, sí que es necesario considerar los riesgos asociados a ésta, tal como se comentará más adelante en el apartado "Evaluación de riesgos y herramientas".

Por ejemplo, las tareas necesarias para planificar y llevar a cabo un programa de formación en la empresa podrían ser las siguientes:

1) Identificar las necesidades de los trabajadores.

2) Planificar el programa de formación.

3) Diseñar y elaborar los cursos o contratar a unos consultores externos para llevarlos a cabo.

4) Convocar a los participantes.

5) Llevar a cabo las actividades de formación.

6) Evaluar a los participantes.

7) Evaluar a los formadores y al programa.

En este caso, sería necesario realizar una división más exhaustiva de estas tareas o fases.

Un proyecto se constituye de un conjunto de tareas (o actividades) determinadas por distintas características como el nombre, la duración estimada, su interrelación con otras tareas del proyecto, los recursos que requiere para ser realizada y una descripción sobre en qué consiste.

3.2.2. Las interrelaciones entre las tareas

Las diferentes tareas de un proyecto no se realizan todas a la vez, ni todas de forma secuencial, una detrás de otra. Hay que analizar qué tareas son dependientes de otras, cuáles son independientes y, entre las que tienen dependencias, cabe priorizar la ejecución en función de la criticidad y de las dependencias entre ellas. Un jefe, es decir, quien tiene la orden de ejecución, ha de ordenarlas en el tiempo en función de los recursos humanos y sus capacidades, de manera que sea posible hacerlas todas respetando el orden y minimizando el tiempo de ejecución y, por lo tanto, también minimizando el coste.

Hacer todas las tareas de manera sucesiva comportaría alargar de modo innecesario el proyecto, y hacerlas todas simultáneamente es imposible, ya que algunas pueden depender de otras (es decir, para empezar una tarea hace falta haber finalizado otras).

Por ello, una vez se han identificado las principales tareas del proyecto, hay que dar un paso más y establecer el encadenamiento más lógico y conveniente entre ellas.

Unas tendrán una prioridad, otras se tienen que hacer en momentos diferentes, unas terceras serán secuenciales y otras se podrán hacer en paralelo.

Por ejemplo, en una construcción de un edificio, hay que finalizar las excavaciones antes de poder iniciar los trabajos de cimentación, y no se suele pintar hasta que los electricistas y los albañiles han finalizado su trabajo.

Estas relaciones se conocen con el nombre de "precedencia y sucesión de las tareas".

Las tareas o actividades de un proyecto no se realizan todas de manera paralela, sino que algunas se harán de manera consecutiva y dependerán de la finalización o el inicio de otras.

3.2.3. La gestión de los recursos

Para muchas personas, la gestión de los proyectos concluye en la planificación de las tareas, tratadas en el apartado anterior. Es decir, en la división del proyecto en tareas y subtareas, en la determinación de la duración de cada una y en la definición de las interrelaciones y vinculaciones existentes entre las distintas partes que componen el proyecto.

Desgraciadamente, sólo con este trabajo no se puede garantizar el éxito y la eficiencia de un proyecto.

Si no se introduce en la planificación del proyecto la gestión y asignación de recursos a cada una de las tareas, la planificación podrá alcanzar uno de los objetivos, que es la consecución de los hitos, pero no se contemplarán aspectos como el control de los costes, la correcta utilización y optimización de los recursos disponibles (personas, maquinaria, materiales, etc.).

Así pues, la realización de las tareas que se han identificado debe estar acompañada de la asignación de los recursos que se han de utilizar en cada una de ellas.

La gestión de los recursos es muy importante en la gerencia de un proyecto por varias razones:

- Los recursos disponibles, humanos, técnicos, financieros, siempre se limitan a una empresa u organización.
- Los tipos de recursos utilizados y su cantidad determinan los costes del proyecto.
- Los recursos y, en especial, los humanos y técnicos son los responsables de llevar a cabo las tareas en el tiempo previsto y con la calidad necesaria.
- Los proyectos exigen el uso de recursos muy variados (máquinas, especialistas en diferentes materias, mano de obra, fungibles, etc.).
- Estos recursos no se concretan de manera estable durante todo el proyecto, sino que para cada tarea se necesitan recursos diferentes en naturaleza y cantidad.

Existen dos aspectos básicos en la gestión de recursos: la previsión y el equilibrio de los recursos.

Por una parte, por *previsión* se entiende que cada actividad, al ser ejecutada, cuente con la garantía de disponer de los recursos necesarios, mientras que, por otra, el equilibrio tiene como requisito la previsión y procura cubrir las necesidades de los recursos utilizados durante el proyecto, de manera que se minimice la existencia de recursos ociosos (o sobrantes) durante la ejecución.

La realización de las tareas que se han identificado en un proyecto sólo se puede conseguir con la asignación de los recursos humanos responsables de su ejecución y los recursos materiales que se han de utilizar en cada una.

Para saber más

En el apartado "El equipo de trabajo" de este mismo módulo se presenta una descripción de los principales roles que pueden participar en un proyecto de sistemas informáticos.

3.2.4. La redistribución de los recursos

Tal como se comentaba en el apartado anterior, uno de los objetivos de la gestión de los recursos consiste en minimizar la existencia de recursos ociosos y, a la vez, garantizar que todas las tareas del proyecto dispondrán de los recursos necesarios en el momento justo.

Así, tal como se ha visto en el apartado "Planificación y seguimiento", una de las principales actividades en la fase de seguimiento es detectar las incidencias y los retrasos de las diferentes actividades del proyecto e intentar corregirlos.

Los cambios de planificación y retrasos provocan también desajustes en la gestión de los recursos, ya que éstos se deben reasignar en función de estos cambios.

Por otra parte, tal como ya hemos mencionado, los recursos suelen ser escasos y limitados, por lo que, a veces, la menor disponibilidad de estos recursos, con respecto a la previsión realizada en la fase de definición, obliga a redistribuir las cargas de estos recursos durante la vida del proyecto.

Una responsabilidad importante del jefe de proyectos y del gerente de recursos es aplicar técnicas y utilizar metodologías adecuadas para la correcta asignación, reasignación y resolución de sobreasignaciones de recursos a las tareas.

3.2.5. Gráficos de tiempo y diagramas

Existen dos diagramas básicos en la gestión de proyectos, el diagrama de Gantt y el diagrama Pert, que tienen la forma siguiente:

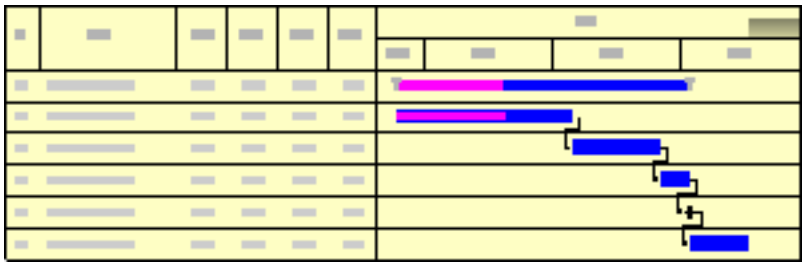


Diagrama de Gantt

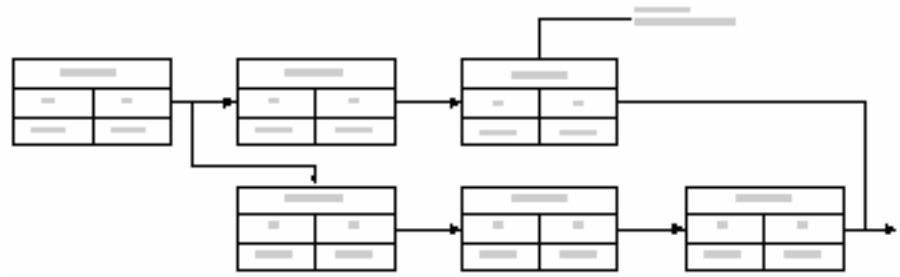


Diagrama Pert

Cada uno presenta una serie de ventajas e inconvenientes, por lo que es el uso de los dos, combinados, la mejor opción para la gestión de proyectos.

Una manera no gráfica de representar las tareas en un proyecto es por medio de tablas.

Un ejemplo podría ser:

Tabla de tareas

Número tarea	Nombre tarea	Duración (en días)
1	Inicio	0

Número tarea	Nombre tarea	Duración (en días)
2	A	4
3	B	3
4	C	2
5	D	2
6	E	2
7	F	4
8	G	5
9	H	4
10	I	2
11	J	2
12	Fin	0

El paso siguiente que se ha de considerar son las interrelaciones entre las tareas, lo que se conoce como precedencia y sucesión de las tareas.

Para saber más

Para ampliar información sobre las interrelaciones entre las tareas, podéis consultar el apartado "Las interrelaciones entre las tareas" de este mismo módulo.

Suponiendo las precedencias siguientes:

- ADH (la tarea D no se puede empezar hasta que se finalice la tarea A, y tras la tarea D está la tarea H).
- BEIJ.
- CFIJ.
- CGH.
- IJ (cuando acabe la tarea I se podrá empezar la tarea J).

Para ejemplificarlo, podríamos encontrar que:

- La tarea A es el hito de inicio del proyecto.
- La tarea D es la recopilación de los datos necesarios para llevar a cabo la definición funcional.
- La tarea H es el diseño funcional de la solución que se ha de implementar.

O, por ejemplo, que:

- La tarea C es la definición de la arquitectura de la infraestructura del sistema (hardware).
- La tarea F es la adquisición del hardware necesario.
- La tarea I es la instalación de este hardware.
- La tarea J es la configuración de las máquinas.

Estas dependencias se reflejarían de la manera siguiente en la tabla:

Tabla con precedencias

Número tarea	Nombre tarea	Duración (en días)	Tarea precedente
1	Inicio	0	
2	A	4	
3	B	3	
4	C	2	
5	D	2	A
6	E	2	B
7	F	4	C
8	G	5	C
9	H	4	D;G
10	I	2	E; F
11	J	2	I
12	Fin	0	

O si se considerara el número de la tarea, en lugar del nombre:

Tabla con precedencias

Número tarea	Nombre tarea	Duración (en días)	Tarea precedente
1	Inicio	0	
2	A	4	
3	B	3	
4	C	2	
5	D	2	2
6	E	2	3
7	F	4	4

Número tarea	Nombre tarea	Duración (en días)	Tarea precedente
8	G	5	4
9	H	4	5; 8
10	I	2	6; 7
11	J	2	10
12	Fin	0	

Siguiendo con el ejemplo, puesto que las tareas A, B y C (2, 3 y 4) no dependen de ninguna otra, empezarán todas a la vez y se podrá considerar que el proyecto ha finalizado cuando se hayan llevado a cabo las tareas H, I y J (9, 10 y 11).

Esta tabla de tareas se podría representar, también gráficamente, con un diagrama de Gantt.

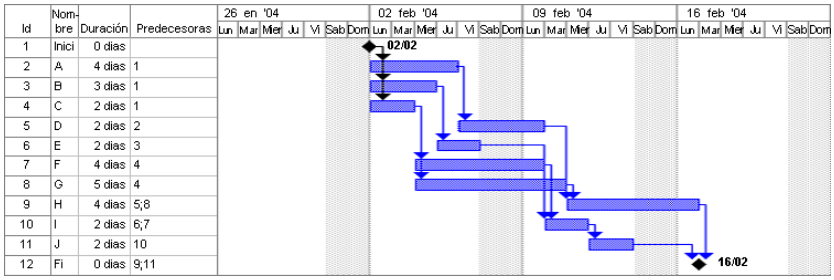


Diagrama de Gantt

Existen diferentes gráficos y diagramas que representan las tareas de un proyecto y sus interrelaciones. Los principales son el diagrama de Gantt, el diagrama de Pert y las tablas de tareas.

3.2.6. Herramientas, métodos y técnicas

Camino crítico

Una de las principales técnicas o métodos de gestión de proyectos es el cálculo del camino crítico.

Un camino es cada uno de los encadenamientos necesarios de tareas, partiendo de las tareas iniciales y siguiendo las sucesiones (precedencias) definidas.

Ejemplo anterior

En el ejemplo del apartado anterior, los diferentes caminos serían ADH, BEIJ, CFIJ, CGH.

Cada uno de estos caminos tiene una duración prevista, que es la suma de los tiempos necesarios para realizar cada tarea.

Al estar encadenadas, la duración total del camino es la suma de sus duraciones.

Siguiendo el ejemplo, las duraciones de los diferentes caminos del proyecto son las siguientes:

- ADH = 4 + 2 + 4 = 10 (el tiempo necesario para llevar a cabo las tareas A, D y H es de 10 días).
- BEIJ = 3 + 2 + 2+ 2 = 9
- CFIJ = 2 + 4 + 2 + 2 = 10
- CGH = 2 + 5 + 4 = 11

Es muy importante, en todo proyecto, detectar el camino crítico, que es el más largo.

Su importancia viene dada por dos factores:

- Todo retraso en una de las tareas del camino crítico provocará que el proyecto, en conjunto, se retrase. Es decir, que finalice más tarde.
- Todo adelanto en la ejecución de una tarea que pertenezca al camino crítico supondrá un adelanto en la ejecución del proyecto.

Ejemplo anterior (continuación)

Según el ejemplo, un retraso de un día en una de las tareas B o E no supondría un retraso en el proyecto:

- ADH = 10
- BEIJ = 10
- CFIJ = 10
- CGH = 11

En cambio, un adelanto (recuperación de tiempo) de dos días en la tarea G provocaría que el proyecto pudiera acabar antes:

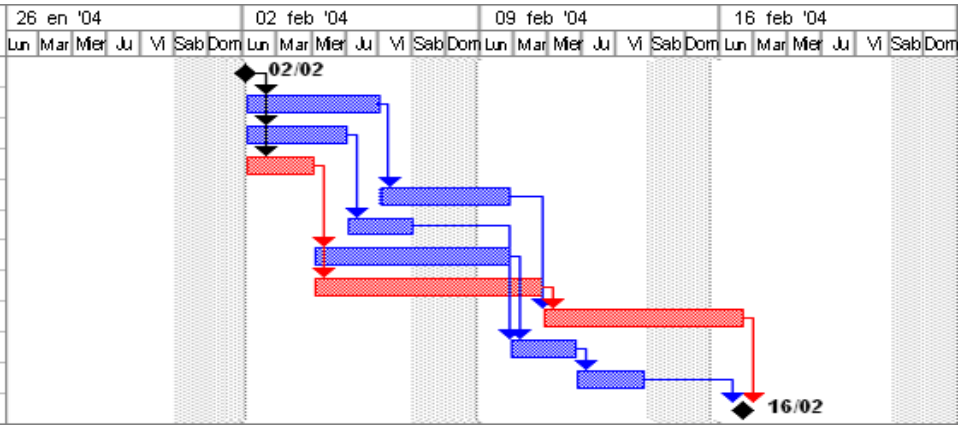
- ADH = 10
- BEIJ = 9
- CFIJ = 9
- CGH= 9

En este caso, además, el cambio de duración de las tareas supone que el camino crítico pase a ser el que forman las tareas ADH.

El camino crítico y las tareas críticas se suelen destacar en los gráficos con el objetivo de llamar la atención de los responsables del proyecto a fin de que maximicen su seguimiento.

Partiendo del ejemplo inicial.

Id	Nombre	Duración	Predecesoras
1	Inici	0 dias	
2	A	4 dias	1
3	B	3 dias	1
4	C	2 dias	1
5	D	2 dias	2
6	E	2 dias	3
7	F	4 dias	4
8	G	5 dias	4
9	H	4 dias	5;8
10	I	2 dias	6;7
11	J	2 dias	10
12	Fi	0 dias	9;11



Los márgenes

Las tareas que no pertenecen al camino crítico se dice que tienen margen, es decir, que su momento de inicio y de fin no se encuentran rígidamente marcados, sino que pueden variar, un margen, en el tiempo, sin que ello afecte a la fecha de finalización del proyecto.

Esta información es muy importante para el jefe de proyectos, ya que cuenta con unos márgenes de tiempo de retraso o adelanto de determinadas tareas con el fin de compensar desviaciones del proyecto, sobreasignaciones de recursos, etc.

A partir de la duración de cada tarea y de la relación con sus predecesoras, estos métodos permiten calcular:

- "tan pronto como" puede empezar una tarea,
- "tan pronto como" puede finalizar una tarea,
- "tan tarde como" puede empezar una tarea,
- "tan tarde como" puede finalizar una tarea.

De manera simplificada se denomina *tan pronto* al tiempo más temprano que una tarea puede empezar o acabar y *tan tarde* al tiempo más tarde admisible para una tarea.

Para ello se utilizan dos métodos de cálculo de las fechas "tan pronto" y "tan tarde" de un proyecto atendiendo a la ordenación "tan pronto" y a la ordenación "tan tarde" respectivamente.

- La ordenación "tan pronto como" determina para cada tarea la fecha de inicio y de fin "tan pronto como", teniendo en cuenta las relaciones con las tareas predecesoras y sus duraciones. En el ejemplo, suponiendo que el proyecto se inicia el día 1, la tarea inicio no tenía duración, por lo que, la tarea A empieza el día 1 y acaba el día 4 y la tarea D empezará el día 5 y acabará el día 6.

- La ordenación "más tarde" calcula para cada tarea la fecha de comienzo "más tarde" y la fecha de finalización "tan tarde" sin cambiar la fecha del proyecto determinada por la planificación "tan pronto". Con el fin de determinar la planificación "tan tarde" se debe empezar por la última tarea del proyecto y "remontar" sucesivamente hasta el inicio.

Una vez calculadas las fechas de inicio y finalización "tan pronto" y "tan tarde" se pueden calcular los márgenes.

El uso correcto de los márgenes es importante para la gestión de los proyectos, ya que las tareas con margen se pueden desplazar en el tiempo, adelantarse o retrasarse, dentro de los márgenes establecidos por la medida del margen sin que esto afecte al plazo del proyecto.

Cálculo de las fechas "tan pronto"

El proyecto se inicia el día 1:

a) Para una tarea con una sola predecesora

- el comienzo "tan pronto" de una tarea es igual a la finalización "tan pronto" de la tarea predecesora + 1. Siguiendo el ejemplo, para la tarea A = 0 + 1 = 1 y para la tarea D = 4 + 1 = 5

- el final "tan pronto" de una tarea es igual al principio "tan pronto" de la tarea + la duración de la tarea – 1. Por ejemplo, el final "más bien" de la tarea D es igual a 5 + 2 – 1 = 6. La tarea acabará el día 6 y por ello se resta una unidad en la fórmula.

b) Para una tarea con diferentes predecesoras

- El comienzo "tan pronto" de una tarea es el valor máximo de las finalizaciones "tan pronto" de las tareas predecesoras + 1. Por ejemplo, para la tarea H = Máx (6, 7) + 1 = 7 + 1 = 8

- El final "tan pronto" de una tarea es igual al principio "tan pronto" de la tarea + duración – 1. Para la tarea H del ejemplo, sería 8 + 4 – 1 = 11.

La tabla completa de las fechas "tan pronto" del ejemplo sería la siguiente:

Número tarea	Nombre tarea	Duración (en días)	Tarea precedente	Inicio "tan pronto"	Fin "tan pronto"
1	Inicio	0		0	0
2	A	4	Inicio	1	4(*)
3	B	3	Inicio	1	3
4	C	2	Inicio	1	2
5	D	2	A	5	6
6	E	2	B	4	4
7	F	4	C	3	6
8	G	5	C	3	7
9	H	4	D;G	8	11
10	I	2	E; F	7	8
11	J	2	I	9(**)	10
12	Fin	0	H; I; J	11	11

(*) La fecha de finalización "tan pronto" es el comienzo "tan pronto" (1) más la duración (4) menos 1 = 4
(**) La fecha de comienzo "tan pronto" es la finalización "tan pronto" de la predecesora (8) más 1 = 9

Cálculo de las fechas "tan tarde"

El proyecto acaba el día 11.

a) Para una tarea con una sola sucesora

- la finalización "tan tarde" de una tarea es la finalización "tan tarde" de la sucesora – la duración de la tarea sucesora. Por ejemplo, para la tarea J sería 11 – 0 = 11 y para la tarea I, 11 – 2 = 9

- el comienzo "tan tarde" de una tarea es la finalización "tan tarde" de la tarea – duración + 1. Como ejemplo, para la tarea I, el comienzo "tan tarde" es 9 – 2 + 1 = 8.

b) Para una tarea con diferentes sucesoras

- La finalización "tan tarde" de una tarea es el mínimo de los comienzos "tan tarde" de las tareas sucesoras – 1. Por ejemplo, para la tarea I = Mín. de los comienzos "tan tarde" (J, Fin) – 1 = Mín. (10, 11) – 1 = 9.

- El comienzo "tan tarde" de una tarea es la finalización "tan tarde" de la tarea – duración + 1. Para la tarea I del ejemplo, sería 9 – 1 + 1 = 8.

La tabla completa de las fechas "más pronto" y "más tarde" será la siguiente:

Número tarea	Nombre tarea	Duración (en días)	Tarea precedente	Inicio "tan pronto"	Fin "tan pronto"	Inicio "tan tarde"	Fin "tan tarde"
1	Inicio	0		0	0	1	1
2	A	4	Inicio	1	4	2	5
3	B	3	Inicio	1	3	3	5
4	C	2	Inicio	1	2	1	2
5	D	2	A	5	6	6	7
6	E	2	B	4	4	6	7
7	F	4	C	3	6	4	7
8	G	5	C	3	7	3	7
9	H	4	D;G	8	11	8	11
10	I	2	E; F	7	8	8	9
11	J	2	I	9	10	10	11
12	Fin	0	H; I; J	11	11	11	11

Se puede comprobar, por ejemplo, que en algunas tareas, los tiempos "tan pronto" y "tan tarde" coinciden, es decir, las fechas "tan pronto" y "tan tarde" de comienzo y final son las mismas, como, por ejemplo, las tareas C, G y H.

En cambio, otras tareas como, por ejemplo, la E, tienen margen, lo que significa que se pueden iniciar en diferentes momentos o se pueden alargar sin alterar el plazo final previsto.

Métodos de redistribución

Los métodos de redistribución de los recursos pretenden conseguir un diagrama de carga tan uniforme como se pueda.

La aplicación de estos métodos de redistribución se puede deber a varias razones:

- Cambios en la planificación de las tareas.

- Incidencias ocurridas durante el proyecto y retrasos.

- Cambios en disponibilidad de los recursos previstos.

- Solapamiento temporal de tareas que requieren el mismo tipo de recurso.

Esta operación de redistribución resulta mucho más complicada de lo que pueda parecer.

Para poder llevarla a cabo, es necesario redibujar el gráfico de Gantt, basándose en la tabla de los tiempos de "Antes y Después" y los márgenes totales que puede soportar cada tarea. De esta manera, se podrá evaluar la posibilidad de realizar las tareas en momentos en los que se disponga de recursos sobrantes (ociosos o sin asignación).

Las tareas que componen el camino crítico no tienen margen y, por lo tanto, si no se dispone de recursos para llevarlas a cabo, es imposible que el proyecto acabe en la fecha prevista.

Por esta misma razón, en la redistribución de recursos se ha de dar prioridad a las tareas críticas.

Una vez redistribuidos los recursos, si continúa habiendo problemas de asignación, es necesario:

- evaluar el posible retraso o adelanto de tareas, lo cual puede implicar un retraso en el proyecto que provocará una desviación en tiempo,

- incorporar nuevos recursos para llevar a cabo estas tareas, lo cual implicará un aumento de los costes previstos que provocará una desviación presupuestaria.

3.3. Proyectos informáticos

El resultado final de un proyecto informático es un sistema informático compuesto de un conjunto de elementos y módulos que procesan y gestionan informaciones y datos, por medio de diferentes métodos, procedimientos y controles.

Un sistema informático, tal como hemos comentado en las aplicaciones del software (podéis ver el apartado "Aplicaciones del software") puede tener unas funcionalidades y estar dirigido a unos entornos muy diversos.

En el apartado anterior hemos presentado los principales conceptos de la gestión de proyectos generales.

Ahora bien, un proyecto informático presenta unas particularidades y características específicas, sobre todo por los elementos que componen un sistema informático:

- Hardware. Dispositivos electrónicos y dispositivos electromecánicos, como la CPU y la memoria (podéis ver el módulo "El hardware"), los circuitos integrados, componentes electrónicos, tarjetas de circuito impreso, etc.

- Software. Son los diferentes programas y estructuras de datos, así como las aplicaciones que se integrarán o utilizarán con los sistemas, como los gestores de bases de datos, los servidores de mensajería, servidores de aplicaciones o Internet, por ejemplo.

- El equipo de trabajo. Compuesto por diferentes profesionales que adoptan un papel o más.

Estas características y particularidades hacen que en la gestión de proyectos informáticos sea necesario tener en cuenta diferentes fases, como el diseño, el desarrollo e integración, la verificación y la implantación, y la documentación necesaria, tanto para explicar el uso y las operaciones del sistema como para dar fe de cómo se ha implementado el sistema para facilitar un posterior mantenimiento.

En los apartados siguientes introduciremos los principales conceptos en la definición inicial del proyecto (evaluación de riesgos, recursos, equipo de trabajo, evaluación de esfuerzos), el diseño de la solución, la gestión de los costes, el desarrollo y la integración, así como la documentación y el mantenimiento del software.

Un proyecto informático presenta unas características particulares, no sólo por las tareas que se han de desarrollar (ingeniería del software), sino también por los elementos que componen los sistemas informáticos: el hardware y el software.

3.3.1. Fases y tareas principales

Todo proyecto informático sigue las mismas fases, aunque en función de su tipología y naturaleza, se llevan a cabo más o menos subfases.

En los apartados siguientes se definirá, con más detalle, cada una de estas fases:

- Diseño de la solución. Tiene como objetivo crear una solución conceptual, identificando la aplicación de las principales funcionalidades, las características necesarias y las adaptaciones que se han de llevar a cabo. También se definen diferentes modelos operacionales para el uso y trabajo del futu-

ro sistema. Asimismo, se diseñan los principales elementos y módulos y se prepara el entorno de desarrollo, la arquitectura del sistema y los procesos de aseguramiento de la calidad del producto. En esta fase se llevan a cabo los procesos siguientes, principalmente:

- inicio del proyecto,
- requerimientos de negocio,
- requerimientos funcionales,
- diseño general,
- diseño detallado,
- preparación del entorno de desarrollo.

- Desarrollo y construcción. El objetivo de la fase de desarrollo y construcción es la creación de los códigos de los componentes, la definición y configuración de las bases de datos, la construcción de los programas de las aplicaciones, la documentación y las pruebas de aseguramiento de calidad. Los dos principales procesos son el desarrollo y la preparación de la implantación.

- Implantación. En esta fase final se configura el entorno de producción, se verifica la solución completa, se ajustan los parámetros finales, se llevan a cabo las pruebas de funcionalidad y se forma a los usuarios de la solución, y se entregan los documentos de uso y trabajo.

Aun así, previamente al diseño se ha de definir el proyecto evaluando los riesgos, los recursos necesarios y definiendo el equipo de trabajo.

Esta planificación definirá los costes necesarios.

Y para obtener esta planificación se ha de realizar una evaluación y estimación siguiendo un conjunto de herramientas y métodos.

Una vez definido el proyecto, es necesario realizar el diseño de la solución, desarrollarla e implantarla.

Evaluación de riesgos y herramientas

Se puede definir el riesgo en la gestión de proyectos como la incertidumbre que implica la ejecución de una estimación. Es decir, la posibilidad de que surjan imprevistos y cambios que no permitan cumplir la planificación tanto en tiempo como en costes, trabajo o disponibilidad de recursos, entre otros.

Todo proyecto supone unos riesgos asociados, más o menos probables y graves, en función de su complejidad.

En todo proyecto informático suele haber diferentes componentes o elementos que pueden suponer un riesgo:

- La misma naturaleza del software. Errores de desarrollo, herramientas poco probadas, incompatibilidades de integración y comunicación entre aplicaciones diferentes, etc.

- Problemas asociados con el hardware. Fallos en las máquinas, errores de integración de sistemas, etc.

- La complejidad del proyecto. Cuanto mayor sea un proyecto y más elementos intervengan en él (número de gente, proveedores, máquinas, aplicaciones, etc.), más probabilidades existen de que se den fallos en el equipo de producción.

- El tamaño del proyecto. Cuanto mayor y más complejo sea el proyecto, más complicado será llevar a cabo su valoración y planificación y, por lo tanto, existirán más probabilidades de error.

Por lo tanto, pues, en todo proyecto informático, en su fase inicial y dentro de la valoración, será necesario evaluar los riesgos y tomar las medidas pertinentes.

Este proceso se puede dividir en tres pasos:

- identificación de los riesgos,

- evaluación de los riesgos,

- actuaciones correspondientes.

Para identificar los riesgos se deben tener en cuenta los aspectos siguientes:

- La descripción final del producto o productos del proyecto.

- La planificación de otras áreas con las cuales interaccione nuestro proyecto.

- La información histórica en el desarrollo de otros proyectos informáticos similares.

En esta primera fase de identificación de los riesgos se deben conseguir los elementos siguientes:

- Relación de posibles fuentes de riesgo. Tienen una probabilidad alta o media de suceder.

- Una lista de riesgos potenciales, que tienen una probabilidad menor de suceder como, por ejemplo, la marcha, de la empresa, de parte del equipo de trabajo o cambios en la tecnología de implementación del software.

- Una lista de síntomas de riesgo.

Una vez recogida e identificada esta información, se ha de averiguar la probabilidad de que ocurra cada uno de estos riesgos y qué consecuencias tendrían para el proyecto.

Para ello se deberán llevar a cabo los pasos siguientes:

- Definir una escala de valores que permita asignar a cada riesgo su probabilidad de que suceda.

- Enumerar las consecuencias de estos riesgos. Es decir, estimar el impacto del riesgo en el proyecto y en el producto o productos finales.

- Identificar en qué momento podría suceder y cuánto tiempo duraría.

Finalmente, el tercer paso, una vez se disponga de estos datos, es decidir qué riesgos se han de prevenir y para qué otros es más rentable asumirlos sin establecer ninguna medida.

En el segundo caso, simplemente, es necesario aceptarlos y desarrollar, si procede, un posible plan de choque por si sucedieran.

Para el primer grupo de riesgos se puede optar por dos alternativas:

- Evitarlos, suprimiendo la causa.
- Si no es posible evitarlos, se deberá reducir la posibilidad de que ocurran.

La gestión de los riesgos en un proyecto consiste en el análisis de las posibles incidencias que puedan surgir, la evaluación de su magnitud, el impacto y las consecuencias que tendrían en el proyecto y los productos que se deben desarrollar y, finalmente, tomar las medidas adecuadas, tanto para evitarlos como para disminuir la probabilidad o, simplemente, aceptarlos.

Recursos necesarios

Durante la primera fase de definición y planificación del proyecto se deberán especificar los recursos necesarios para llevarlo a cabo.

Tal como ya hemos comentado en el apartado anterior, se habrá de definir qué recursos son necesarios, cómo se pueden conseguir (contratación, miembros de la organización, alquiler, etc.), cuándo se necesitarán y durante cuánto tiempo.

La diferencia, sin embargo, tal como ya hemos comentado anteriormente, es que se deberán tener en cuenta dos tipologías especiales de recursos: los recursos de software y de hardware.

Para saber más

Para tener más información sobre el tipo de recursos y su gestión podéis consultar el apartado "La gestión de los recursos" de este mismo módulo.

- Los recursos de hardware, que se necesitan tanto para desarrollar el proyecto (servidores de desarrollo, máquinas de verificación, etc.), como los que llevará o los que formarán parte del producto final (infraestructura de producción, terminales, dispositivos para los usuarios, etc.).

- Recursos de software, que engloban tanto los lenguajes de desarrollo, los compiladores o intérpretes correspondientes, las herramientas de apoyo al diseño y producción, como el software resultante del proyecto (programa y/o aplicaciones que formarán parte del nuevo sistema informático).

Para saber más

Para tener más información sobre los lenguajes de desarrollo podéis consultar el apartado "Principales lenguajes de desarrollo". Para tener más información sobre compiladores o intérpretes podéis consultar el apartado "Traductores". Para tener más información sobre las herramientas de apoyo al diseño y producción, podéis consultar el apartado "Herramientas CASE". Todos estos apartados los encontraréis en este mismo módulo.

En un proyecto informático no sólo se han de considerar los recursos humanos y materiales, sino también los recursos de software y de hardware.

Sólo como introducción, enumeraremos a continuación algunos de los recursos de software que pueden utilizarse y ser necesarios para llevar a cabo el proyecto:

- Herramientas de análisis y diseño que permiten crear los modelos de software con el fin de validar la consistencia y la validez del diseño, lo que hace disminuir la existencia de posibles errores de desarrollo.

- Herramientas de simulación y creación de prototipo. Facilitan que tanto el equipo de trabajo, como, sobre todo, el usuario final del sistema puedan evaluar y entender el funcionamiento del futuro sistema antes de desarrollarlos.

- Herramientas de estructura que permiten la definición y gestión de las bases de datos y fuentes de información (herramientas de gestión documental, de contenido, etc.).

- Herramientas de desarrollo: editores, compiladores, depuradores, lenguajes de desarrollo, lenguajes de acceso y gestión de bases de datos, etc.

- Software ya existente, bien sean productos estándar, del mercado o librería de componentes desarrollados anteriormente, que se pueden reutilizar en el proyecto.

- Herramientas de prueba que facilitan no sólo la tarea de verificación tanto del producto final, como de los pasos intermedios, sino también la integración de los diferentes componentes del sistema informático futuro.

Entre los recursos de software es necesario tener en cuenta no sólo el producto final, sino también los recursos necesarios para desarrollarlos (herramientas de análisis y diseño, simulación, lenguajes de desarrollo, librerías existentes o herramientas de verificación).

La reutilización del software es muy importante en el desarrollo de un proyecto; el hecho de utilizar módulos o partes ya desarrollados por otros proyectos hace que el tiempo de proyecto sea más corto y se agudiza la fiabilidad, ya que son partes ya probadas por el usuario. Un alto nivel de reos indica experiencia, fiabilidad y menor coste.

Evaluación y estimación

Con el fin de poder establecer los recursos necesarios para llevar a cabo el proyecto, se ha de realizar una valoración, es decir, se debe estimar la "magnitud" del proyecto.

Para este proceso, se ha de disponer, por una parte, de información histórica de los anteriores proyectos similares realizados y, por otra, contar con la experiencia de los profesionales y expertos de la compañía.

Asimismo, hay un conjunto de indicadores y métricas de productividad del software que permiten dar apoyo a esta tarea:

- Líneas de código. Es uno de los indicadores más comunes a la hora de planificar el esfuerzo necesario.

- Puntos de función o funcionalidades que se han de desarrollar en el sistema informático.

- Etc.

Para realizar esta tarea, los analistas también disponen de un conjunto de herramientas en el mercado que implementan técnicas de estimación que aplican fórmulas matemáticas con parámetros obtenidos a partir de la información histórica.

Uno de los modelos más utilizados es el COCOMO (*constructive cost model*). Este modelo calcula el número de personas y el tiempo de gestión necesario para finalizar un proyecto a partir de:

- el tamaño del proyecto, es decir, líneas de código,
- el tipo de software que se ha de desarrollar,
- y la organización y experiencia del equipo de trabajo.

Gestión del coste

La estimación de los costes se realiza, por una parte, a partir de los requerimientos del proyecto y la información histórica, pero, sobre todo, a partir de la planificación de los recursos necesarios definidos (equipo de trabajo, recursos materiales, recursos de software y recursos de hardware).

Por ello es importante realizar el cálculo de los costes una vez evaluada la planificación y las necesidades de recursos dentro de la fase de definición inicial del proyecto.

Esta estimación debe ser una valoración cuantitativa del coste necesario con el fin de disponer de los recursos necesarios para llevar a cabo el proyecto en el tiempo previsto.

Durante todo el proyecto se deberá realizar un seguimiento del presupuesto inicial para verificar que se está cumpliendo y, en caso contrario, se deberá actualizarlo y tomar las medidas necesarias.

Este control del coste implica vigilar el gasto que se va produciendo con el fin de detectar variaciones sobre la estimación, registrar los cambios que se produzcan e informar a los implicados de los cambios autorizados.

Es necesario que la planificación del proyecto se evalúe y defina el presupuesto inicial con el fin de poder llevar a cabo un seguimiento durante la ejecución del proyecto, así como tomar las medidas necesarias en caso de desviación.

El coste de un proyecto tiene dos grandes partidas:

- Coste del hardware y coste del software.

- Coste de las horas de desarrollo.

El primero de los dos costes no acostumbra a tener grandes desviaciones; el segundo, en cambio, fluctúa mucho y es el que, si no se ha hecho una planificación esmerada, puede llevar a grandes desviaciones que hagan entrar en pérdidas al proyecto.

3.3.2. El equipo de trabajo

Tal como se ha detallado en el apartado de "Planificación y seguimiento", a continuación detallamos los principales roles que pueden participar en un proyecto de sistemas informáticos.

La pretensión es enumerar y describir la mayoría de los roles de los miembros que pueden participar en proyectos de estas características.

Con el fin de describir estos diferentes roles de los participantes de un equipo de proyectos, se agrupan en diferentes categorías o familias:

- Gestión del proyecto.
- Procesos y negocios.
- Marketing.
- Diseño gráfico y multimedia.
- Ingeniería.
- Control de la calidad.

Dependiendo de la tipología del proyecto, el equipo de trabajo estará formado por unos roles u otros.

En un proyecto una persona puede asumir más de un rol, por ejemplo, puede ser que un miembro del equipo adopte el papel de jefe de proyecto con una dedicación del 50% de su tiempo y la otra parte de su tiempo actúe como analista funcional, en la fase de diseño y como verificador, en la fase final de verificación; o puede ser, por ejemplo, que una misma persona asuma los papeles de analista funcional, en la primera fase, y de analista programador en la siguiente.

Analista programador

Persona que se encarga de analizar el problema que se desea resolver mediante la creación de una aplicación de software específica y que debe determinar la manera exacta como lo solucione este programa.

Ejemplo web

Por ejemplo, en un proyecto de implantación de una solución web, será necesario que en el equipo haya diseñadores gráficos y desarrolladores de webs, mientras que en un proyecto de implantación de una herramienta de gestión financiera se deberá contar con la participación de un experto en la parametrización de aquella herramienta y, seguramente, de un consultor de finanzas y contabilidad, entre otros.

Según la tipología y magnitud del proyecto, el equipo de trabajo estará formado por más o menos papeles, según las necesidades. Aun así, estos roles se pueden agrupar en seis tipos o familias.

Gestión del proyecto

Dentro de esta familia, se identifican entre otros, tres papeles:

- **Gerente del proyecto**. Asigna los recursos, establece las prioridades, coordina la interacción y mantiene al equipo focalizado en el objetivo correcto, y establece las prácticas que aseguran la integridad y calidad de los artefactos del proyecto.

- **Administrador del proyecto**. Lleva el control de las horas y del presupuesto del proyecto, elabora los estados de cuentas y la facturación, programa y controla el pago a proveedores y da apoyo a los miembros del equipo con los gastos de viaje, la entrada de horas, localización, etc.

- **Gestor del conocimiento**. Es el responsable de revisar la documentación que se genera en el proyecto, hacer cumplir los estándares de documentación y revisar que la documentación generada se adecue a los acuerdos definidos en el proyecto.

A menudo, estos tres roles se unifican con una sola persona bajo la denominación de jefe de proyecto o *project management*.

Procesos y negocios

Bajo esta familia de responsabilidad, se agrupan los papeles siguientes:

- **Consultor de negocios**. Realiza el análisis de la industria y de la empresa y la estrategia de negocio del cliente. Asegura la dirección adecuada de la estrategia de posicionamiento. Lidera el análisis de requerimientos, el diseño organizacional del cliente, su posterior formación y su apoyo operacional. Asegura la realización de todos los requerimientos por parte del equipo de desarrollo e implantación del proyecto.

- **Consultor de procesos**. Es el responsable de realizar el análisis de los procesos de negocio de cada una de las áreas operativas del cliente que serán afectadas por el proyecto (por ejemplo, la gestión de los pedidos, la aten-

ción posventa a los clientes, los procesos de facturación, etc.) detallando cada uno de los procesos del cliente en flujos de trabajo y elaborando el caso de negocio correspondiente con las funcionalidades descritas.

- **Consultor de catálogos**. Da apoyo a la selección y clasificación de los productos y servicios del cliente. Define los estándares técnicos y de presentación del contenido de catálogos. Da apoyo al proceso de integración de proveedores. Asegura la instalación y el acceso a las herramientas para la carga de catálogos en el ambiente de pruebas (verificación) y el de producción.

- **Líder de transacciones**. Define las reglas de negocio para las transacciones desarrollando los diagramas de flujo de procesos, generales y detallados por las transacciones de negocio.

Estos roles son adaptados por el analista funcional, con la ayuda de los expertos en la materia que pueda aportar la empresa que pide el proyecto.

Marketing

Dentro de este grupo se engloban todos aquellos profesionales responsables de considerar la gestión de la marca del cliente, el análisis del mercado, el tratamiento de la imagen corporativa, etc.:

- **Consultor de estrategia de marca**. Combina la estrategia del negocio de Internet con la gestión de marca para asegurar la creación de experiencias de marca relevante al público objetivo y desarrolla recomendaciones de estrategia de posicionamiento.

- **Consultor-analista de marketing**. Prepara los presupuestos, las previsiones y los informes de los procesos y actividades de marketing para el cliente. Realiza el análisis detallado de las actividades de mercado y contribuye al desarrollo de la estrategia de marketing del proyecto Internet, en caso de la implantación de un sitio de Internet.

- **Consultor de marketing**. Es el responsable de buscar oportunidades estratégicas en el mercado utilizando la información proporcionada por el cliente y cualquier otra fuente fiable; presenta, posteriormente, sugerencias para la planificación de la estrategia.

Diseño gráfico y multimedia

La parte gráfica de todo proyecto resulta cada vez más importante. Hubo un salto cualitativo de las aplicaciones *host* o servidor a las aplicaciones y programas Windows, pero el gran salto ha sido con el desarrollo de aplicaciones Internet y webs. Existe todo un conjunto de papeles asociados a estas tareas:

- **Experto funcional creativo**. Coordina la imagen del proyecto, ofrece soluciones y coordina al equipo creativo asesorando y guiando al cliente en las decisiones de imagen.

- **Experto funcional de arte**. Responsable de la dirección artística del proyecto, de establecer consistencia en el diseño y de proveer de experiencias e ideas creativas al resto de equipo.

- **Diseñador gráfico**. Responsable de los elementos gráficos

- **Desarrollador de la interfaz gráfica de usuario**. Programa los elementos que compondrán el proyecto, tanto si es una aplicación web como Windows, y optimiza estos elementos.

Ingeniería

El equipo de ingeniería o técnico es el equipo propiamente responsable del desarrollo de todo el software, la instalación y configuración del hardware y la integración con la parte de comunicaciones y redes:

- **Director técnico**. Comprueba que toda la documentación de requerimientos, análisis y diseño esté completa, correcta y firmada por el cliente. Dirige el desarrollo desde el inicio hasta la finalización con éxito. Coordina el equipo de desarrollo, pruebas, QA. Realiza el seguimiento y coordina las actividades relacionadas con proveedores tecnológicos.

- Obtiene requisitos técnicos del cliente para desarrollar la estrategia tecnológica. Asegura la escalabilidad de la solución y sirve como punto de contacto del cliente para cuestiones técnicas.

- **Administrador de contenido**. Desarrolla las estrategias y los procesos para la gestión de contenido. Adapta los servidores de gestión de contenido estándar del mercado a las necesidades del cliente y forma al cliente en el uso de estas herramientas.

- **Líder técnico de contenido**. Define el esquema y diseña el ciclo de trabajo de contenidos. Ayuda a la integración del servidor de gestión de contenidos con el resto del sitio.

- **Arquitecto de información**. Analiza a los usuarios y las tareas que deben llevar a cabo, diseña la arquitectura del programa y la interfaz de usuario, incluyendo la organización de la información y las reglas de navegación. Recoge los objetivos de negocio, los requisitos del usuario, el contenido y la funcionalidad. Organiza el contenido y desarrolla categorías, esquemas de nombres y jerarquías de navegación. Finalmente, crea y documenta casos de uso.

- **Desarrollador del *back-end***. Es el responsable de creación de los accesos a la información (bases de datos, páginas HTML, etc.).

- **Desarrollador de la lógica de negocio**. Construye los componentes de la capa lógica de negocios en una aplicación de tres capas.

- **Desarrollador *front-end***. Es el responsable de programar las páginas y componentes de interfaz de usuario.

- **Ingeniero de software**. Define y desarrolla las clases, los paquetes y los subsistemas que formarán parte del sistema informático.

- **Ingeniero de comunicaciones**. Evalúa los productos existentes y coordina la instalación, la integración, el desarrollo y la configuración de los productos seleccionados.

- **Diseñador de datos**. Participa con el arquitecto de software en la definición de las bases de datos de las aplicaciones y su interacción. Diseña la estructura de datos verificando y garantizando la integridad, la normalización, la optimización, la migración y la integración de datos.

- **Administrador de bases de datos (DBA**). Instala, genera y mantiene la base de datos.

- **Programador de bases de datos**. Crea los objetivos de bases de datos necesarios para la solución.

- **Arquitecto de red**. Diseña la arquitectura de red de los diferentes ambientes del proyecto (producción, calidad, verificación, desarrollo, etc.).

- **Administrador de sistemas**. Administra los sistemas operativos y aplicaciones entre los diferentes ambientes (producción, desarrollo, QA).

Todos estos roles se reparten en tres perfiles:

- Analistas técnicos.
- Programadores.
- DBA.

Aseguramiento de la calidad

- **Responsable de calidad**. Determina la carga de trabajo para el equipo de aseguramiento de la calidad según el tipo de proyecto. Genera el plan de pruebas con los requerimientos del proyecto en desarrollo y dirige la ejecución de las pruebas.

- **Verificador de código**. Revisa los estándares de codificación y el plan detallado de desarrollo del proyecto con el fin de verificar que el código de la aplicación cumpla los estándares antes de entregar el producto al cliente.

- **Verificador de seguridad y acceso**. Verifica la seguridad de la aplicación, lo cual incluye pruebas que midan la fiabilidad del acceso y la seguridad.

- **Verificador de funcionalidad**. Genera, de acuerdo con los escenarios que el responsable de calidad ha definido en el plan de pruebas, las secuencias (*scripts*) y los casos de prueba para identificar errores de funcionalidad y requisitos. Es el responsable de asegurar que la aplicación cumpla con la funcionalidad de acuerdo con los requisitos definidos por el cliente.

- **Verificador de "Look & Feel"**. Comprueba que el diseño gráfico de la solución cumple los requerimientos de Look & Feel antes de su entrega final al cliente.

El trabajo de test se hace entre:

- Jefe de proyecto.
- Analistas.
- Usuarios finales.

3.3.3. Diseño

El diseño es la primera fase de implementación de todo proyecto.

Una vez definidas las necesidades y los requerimientos del sistema informático e identificados los recursos necesarios y formado el equipo de trabajo, es necesario llevar a cabo el diseño del producto que se quiere desarrollar.

Tal como hemos comentado, el diseño parte de las necesidades del sistema y del modelo de interrelaciones de los elementos que deben intervenir.

El diseño es clave en todo proyecto, ya que un mal diseño conlleva errores en el desarrollo.

El primer factor que se ha de tener en cuenta en el diseño de todo sistema informático es la fragmentación del problema, es decir, definir los diferentes componentes que lo formarán, diseñarlos por separado y después diseñar la integración.

Este proceso se conoce como el diseño de la arquitectura.

Diseño de la arquitectura

Nombre que se utiliza para definir un conjunto de componentes hardware y software estandarizados con los que está diseñado un sistema informático: procesador, sistema operativo, conectores, etc.

De esta manera, el diseño se suele dividir en módulos que se comunicarán e integrarán entre ellos para formar el sistema final.

En la fase del diseño también se deben definir la infraestructura necesaria y la arquitectura de software que lo soportará (servidores de Internet y aplicaciones, bases de datos, sistemas de mensajería, sistemas operativos, etc.).

Otra parte importante en la fase de diseño es el diseño de datos. De este diseño dependen la organización, los métodos de acceso o las alternativas de procesamiento de los datos y la información que ha de gestionar el sistema informático.

Como resultado de este diseño, se definen las estructuras de datos.

Finalmente, una parte cada vez más importante en el diseño de soluciones informáticas es la interfaz de usuario, es decir, el diseño del aspecto "visual" de la aplicación. Para ello se han de tener en cuenta aspectos tanto de diseño gráfico, como de usabilidad y navegación.

En la fase de diseño de un proyecto informático se deben definir tanto la arquitectura del sistema y las infraestructuras necesarias, como los módulos y funcionalidades que se han de desarrollar e integrar, pero también las estructuras de datos y el diseño gráfico, usabilidad y navegación de la solución.

Asimismo, es necesario definir, en esta fase, las pruebas de verificación que se llevarán a cabo durante el desarrollo, la integración y la implantación de la solución.

Para llevar a cabo estas definiciones y estos diseños, los analistas informáticos cuentan con diferentes métodos y herramientas.

El objetivo de utilizar un método para el diseño es garantizar la obtención de un software comprensible en el que se puedan detectar los errores con facilidad y que permita el mantenimiento y la extensión posteriores.

Para diseñar el software, conviene disponer de un buen sistema de notación que permita entender el diseño de una manera simple, más fácilmente que leyendo el código de un lenguaje de programación.

Las notaciones ayudan a obtener un detalle de las funcionalidades del sistema que se ha de desarrollar de la manera más clara y comprensible.

Algunas de las principales notaciones que se utilizan son los niveles de abstracción, los diagramas de flujo, las tablas de decisión y los casos de uso.

El objetivo de este material no es profundizar ni detallar estas técnicas, métodos y notaciones, sino sólo destacar la importancia y el uso en el diseño de los sistemas informáticos.

Para poder diseñar, de manera comprensible, los sistemas informáticos y las funcionalidades que se han de desarrollar se utilizan diferentes métodos, herramientas y notaciones.

3.3.4. Desarrollo e integración

La fase de desarrollo empieza después de la aprobación formal de los requisitos del sistema por parte del usuario o cliente y de las fases de diseño de alto nivel (análisis funcional) y diseño detallado (análisis orgánico).

Análisis funcional

Nombre con el que se conoce la primera fase del desarrollo de una aplicación que consiste en reunir todos los datos disponibles del problema que se quiere solucionar y crear un diagrama de flujo o un pseudocódigo con los elementos generales del programa que se creará para solucionarlo.

Análisis orgánico

Fase del diseño de una aplicación en la que se convierte el algoritmo escrito en forma de pseudocódigo o diagrama de flujo en código que pueda entender el compilador, intérprete o ensamblador correspondiente.

A partir de la arquitectura definida del software y del sistema, se han de desarrollar las diferentes funciones necesarias y los componentes de software e implementar los flujos de datos y de control entre estos componentes.

La arquitectura de software se descompondrá en unidades de menor complejidad y a menudo se adopta una metodología de tipo *top-down*.

Para cada elemento o componente del software se habrán indicado, durante la fase de diseño, los datos de entrada, las funciones que realizará y los datos de salida.

Como resultado de la fase de desarrollo, o programación, se obtiene, aparte del código del programa, el documento de diseño detallado y el manual de usuario del software.

Todo módulo desarrollado será objeto, posteriormente, de un test de integración.

Para saber más

Para tener más información sobre los tests de integración, podéis consultar el apartado "Verificación y pruebas" de este mismo módulo.

El software codificado y validado está preparado para entrar en la etapa de transferencia, que se instala sobre la plataforma (hardware) final con el fin de poder llevar a cabo los tests de aceptación especificados.

La fase de desarrollo e integración es de las más complejas, tal como ya se ha visto, por la diversidad de roles del equipo de trabajo que puede participar en ella.

Para saber más

Para tener más información sobre la diversidad de roles del equipo podéis consultar el apartado "El equipo de trabajo" de este mismo módulo.

El objetivo de este material no es describir ni profundizar en el desarrollo de aplicaciones, sino simplemente indicar su importancia dentro del proyecto.

Para saber más

Para tener más información sobre el desarrollo de aplicaciones podéis consultar la unidad "Desarrollo de software" de este mismo módulo.

3.3.5. Verificación y pruebas

En el proceso de verificación y pruebas, además de asegurar que el producto o sistema cumple correctamente las funciones que el cliente ha solicitado, se ha de comprobar que los requerimientos establecidos son los adecuados.

Por este motivo, en esta etapa se debe definir una estrategia de pruebas que ha de considerar:

- la planificación de la prueba,
- el diseño de casos de prueba,
- la ejecución de estas pruebas,
- y la evaluación de los resultados.

La verificación de un sistema informático tiene en cuenta tanto las pruebas de software, como posteriormente las pruebas del sistema.

Existe, en el mercado, además, un conjunto de herramientas y aplicaciones que facilitan la tarea del equipo de pruebas.

Pruebas del software

A la hora de verificar y buscar posibles errores en el software desarrollado hay dos técnicas de verificación:

- **Pruebas de caja blanca**: comprueban que los elementos del software encajan correctamente y que los procedimientos internos cumplen las especificaciones. Se centran en la estructura de control del programa, e intentan buscar errores en el código. Se ponen a prueba los caminos independientes de cada módulo, las iteraciones, las estructuras internas de datos y las decisiones lógicas.
- **Pruebas de caja negra**: comprueban que las funciones que se ejecutan en el software son operativas y sirven para conseguir la función por la que han

sido desarrolladas. Confirman que el software responde bien a las entradas, tanto válidas como no, que las salidas son correctas, y que las funciones son operativas. Intentan encontrar errores, tanto errores funcionales de partes que no cumplen con las especificaciones iniciales, como errores en funciones y estructuras de datos incorrectos, errores en el acceso a bases de datos externas, en la interfaz, de rendimiento, etc.

Con el fin de simplificar y garantizar el proceso de pruebas, se inician en los niveles más básicos, los módulos. Una vez verificado el funcionamiento correcto de los módulos, se comprueba el funcionamiento conjunto de los diferentes módulos que componen una parte del sistema y, finalmente, se verifica el correcto funcionamiento de todo el sistema. De esta manera, las pruebas se van llevando a cabo a partir de la integración de los módulos y partes del sistema.

De esta manera, la localización de los errores es más fácil.

Los pasos que se siguen en el proceso de prueba del software son los siguientes:

- Prueba de las funciones y módulos. Se utilizan pruebas de caja blanca, examinando, entre otros elementos, la interfaz del módulo, las estructuras de datos locales y sus funcionalidades. Las llevan a cabo los miembros del equipo de desarrollo.

- Pruebas de integración. En estas pruebas se utilizan las técnicas de caja blanca y caja negra. Se comprueba que cada módulo funciona bien con los demás garantizando que el conjunto mantiene las funcionalidades previstas. Las llevan a cabo los miembros del equipo de desarrollo, pero pueden intervenir los usuarios y otras personas expertas en las funcionalidades pedidas a las especificaciones.

Una vez realizadas las pruebas de funcionalidad e integración del software, se realizan dos pruebas más: la prueba alfa y la prueba beta.

- La prueba alfa: el usuario final del software lo prueba en el mismo lugar donde se ha desarrollado, en un entorno, por lo tanto, controlado.

- La prueba beta: el usuario prueba el software en el lugar donde será utilizado.

Durante todo el proceso de testeo y pruebas, se van detectando errores que se tienen que corregir y volver a probar; es un proceso con un alto contenido de realimentación y muy delicado, ya que soluciones a errores pueden producir errores en funciones ya probadas con éxito y provocar el desconcierto y desencanto del futuro propietario del software.

Beta tester

Término inglés que se traduciría por "verificador de beta". Se utiliza para referirse a aquellas personas que se encargan de probar el funcionamiento de un programa que todavía no ha salido al mercado (se encuentra en versión beta) con el objetivo de encontrar errores de funcionamiento. De estos errores se informa a la empresa autora del programa para que los corrija con el fin de que la versión definitiva sea tan perfecta como sea posible.

Con el fin de verificar el correcto desarrollo y la funcionalidad del software hay dos técnicas: la caja blanca y la caja negra.

El proceso de verificación también se conoce, en argot informático, depurar.

Depurar

Palabra utilizada por los programadores para referirse a la acción de revisar un programa en desarrollo con el objetivo de encontrar y eliminar posibles errores de funcionamiento que pueda haber (conocidos por *bugs*). La depuración también persigue la optimización del programa para que su funcionalidad y velocidad sean lo mejores posibles. La depuración normalmente se realiza con la ayuda de programas especializados en esta tarea, lo que facilita el trabajo del programador y acorta el tiempo utilizado en la fase de depuración. Para evitar los errores en el funcionamiento de los programas, se suelen lanzar varias versiones beta de cada programa que prueban los especialistas.

Pruebas del sistema

Finalmente, una vez verificado el desarrollo correcto del software, se deben realizar las pruebas del sistema.

En este momento, el software se integra con el resto del sistema (hardware, servidores, etc.).

En estas pruebas participan los responsables de desarrollo de cada elemento y parte del sistema.

Las principales pruebas del sistema que se llevan a cabo son las siguientes:

- **Pruebas de recuperación**. Evalúan si la recuperación de errores es la apropiada. Es decir, se comprueba si la reinicialización, los mecanismos de recuperación de estado del sistema, la recuperación de datos, etc. son los adecuados.

- **Pruebas de seguridad**. Tratan de garantizar que el sistema se encuentra protegido contra robos de clave de acceso, ataques con software, bloqueos del sistema, errores provocados por intentos de ataque al sistema, etc.

- **Pruebas de resistencia**. Intentan garantizar que el sistema tiene unos límites correctos para aguantar en condiciones extremas, por ejemplo, el acceso concurrente de muchos usuarios al sitio web, la disponibilidad de memoria suficiente en el caso de ejecución de muchos procesos o que el ancho de banda de la Red permite garantizar el acceso y el trabajo correctos de los usuarios.

Las pruebas del sistema se dividen en recuperación, seguridad y resistencia.

Herramientas de apoyo

Con el fin de que el proceso de verificación sea lo más rápido posible, hay un conjunto de herramientas en el mercado que permiten disminuir el esfuerzo y el coste de esta fase.

Estos programas se pueden clasificar en función del tipo de ayuda que proporcionan:

- **Generadores de datos de prueba**. Producen datos que hacen que los programas y aplicaciones que se deben verificar se comporten de una manera determinada.

- **Comparador de archivos**. Trabajan comparando diferentes conjuntos de datos de salidas de distintas pruebas.

- **Simuladores**. Permiten imitar el entorno real del software y el sistema informático que se está verificando.

Hay distintas herramientas en el mercado para ayudar al equipo de pruebas y aseguramiento de la calidad del sistema. Estas herramientas se pueden clasificar en generadores de datos de prueba, comparadores de archivo y simuladores.

3.3.6. Operación y mantenimiento

Muchos equipos de trabajo tienden a creer que su misión ha acabado cuando el software, y el sistema informático en general, está instalado y funcionando en las instalaciones del cliente. Pero en la práctica pocas veces es así.

Las peculiaridades de un sistema informático, vista la complejidad de todos los elementos que intervienen en él (podéis ver "Proyectos informáticos"), provocan que se deba supervisar el funcionamiento correcto durante mucho tiempo después de haber sido entregado.

Un requisito de disponibilidad o de fiabilidad del sistema, por ejemplo, es muy difícil garantizar que el programa se comportará adecuadamente en circunstancias no previstas o transcurrido un tiempo determinado, por lo que los tests de aceptación suelen incluir, a petición del cliente, pruebas a largo plazo del software.

Esta fase de supervisión (y corrección de los vicios o defectos del producto) recibe el nombre de *fase de operación*.

Sólo después de que haya transcurrido este intervalo, el cliente acepta definitivamente el software, provisionalmente aceptado al acabar la fase de transferencia.

Al cabo del tiempo, el software definitivamente aceptado se debe poder cambiar, bien como consecuencia de errores no detectados, bien como respuesta ante nuevas necesidades del usuario.

Esta fase siguiente recibe el nombre de *fase de mantenimiento*.

Durante la fase de operación y mantenimiento se genera y actualiza el documento de historia del proyecto, que recoge todos los errores y todas las modificaciones realizadas sobre el software y permite calcular y analizar la fiabilidad del software y el rendimiento del equipo de trabajo (para futuros proyectos).

Una vez implantado el sistema, se ha de iniciar una fase de supervisión, llamada *de operación*, durante las primeras semanas o meses, e iniciar posteriormente la fase de mantenimiento si se deben efectuar modificaciones o adaptaciones a la solución.

3.3.7. Documentación

Durante el ciclo de vida de cualquier proyecto se genera mucha documentación relativa, de tipo técnico, de gestión o documentación complementaria.

La gestión correcta de toda esta documentación es vital, no sólo para el seguimiento del proyecto, sino también para garantizar una información histórica, tanto para llevar a cabo futuros cambios y el mantenimiento, como para contar con información en la que basarse en nuevos proyectos similares.

En proyectos complejos, de larga duración o donde intervengan equipos numerosos de personas (o diferentes empresas), su gestión evita duplicar innecesariamente los documentos generados y permite que los integrantes del equipo de trabajo sepan qué documentación existe, cuál es la última versión, dónde se encuentra localizada, etc.

La gestión de la documentación de un proyecto supone la realización de las tareas siguientes:

- redacción de los procedimientos de gestión de documentación y asignación de referencias a documentos,

- asignación de referencias a los documentos, en la medida en que se generan,

- inclusión de los nuevos documentos en la base de datos (listado) de documentación del proyecto,

- difusión en el equipo de trabajo,

- incorporación de la base de datos del proyecto a la base de datos general de la empresa al finalizar el proyecto.

Algunos de los tipos más frecuentes de documentos de un proyecto son:

- Documentación técnica. Requerimientos técnicos y especificaciones, documentos de diseño, notas técnicas, planos, diagramas, listado de programas, manuales técnicos, de instalación de usuario, etc.

- Documentos de gestión. Presupuestos y ofertas, minutas y acciones, informes de situación, balance de horas y gastos, etc.

- Documentos complementarios. Listas de documentos, listas de productos, presentaciones, planes de calidad, procedimientos, faxes, comunicaciones internas, correos electrónicos, etc.

La gestión de la documentación, que en pequeños proyectos suele ser parte de la tarea del gestor del proyecto, se puede delegar en otras personas en proyectos complejos o en los que el volumen de la documentación generada lo aconseje.

Una vez cerrado el proyecto, la documentación se clasificará en reutilizable y no reutilizable.

La documentación reutilizable se incorporará inmediatamente al archivo de documentación de la empresa, mientras que la no reutilizable se puede destruir una vez transcurrido un período prudencial de tiempo (que incluya garantías, servicios posventa o períodos de posible contratación de una continuación del proyecto).

La gestión de la documentación de un proyecto supone la realización de un conjunto de tareas, como son la redacción de los procedimientos de gestión de documentación y la asignación de referencias, la inclusión de nuevos documentos en la base de datos de documentación del proyecto, su difusión entre el equipo de trabajo y la incorporación de la base de datos del proyecto en la base de datos general de la empresa, cuando éste finalice.

En todo proyecto tienen un interés especial las ayudas automáticas de las que pueda disponer el equipo, entendiendo por esto los programas informáticos destinados a facilitar el trabajo de ayudar a construir software.

Tal como ya hemos comentado, estas ayudas pueden facilitar una actividad de análisis, diseño o implementación propias del procedimiento recomendado por la metodología.

También pueden ayudar a gestionar la organización del proyecto, en particular a controlar el proceso de avance y realización, así como facilitar la tarea de elaborar la documentación exigida por la metodología.

4. Licencias de software

La compra de una licencia de un producto o aplicación representa para el comprador el derecho de instalar y utilizar un programa.

Por cada aplicación de software que se utiliza, el fabricante otorga una licencia de uso que se documenta en el contrato de licencia de usuario.

El software está protegido por la ley de derechos de autor, que establece que un producto no se puede copiar sin la autorización del propietario de los derechos de autor.

Una licencia de software aplica a cualquier tipo de programa (podéis ver el apartado "Aplicaciones del software"): sistemas operativos, procesador de textos, antivirus, servidor de correo, gestor de proyectos, etc.

Estas licencias se pueden clasificar, de manera simplificada, en tres tipos:

- personal,
- servidora,
- de cliente.

Una licencia de software personal implica el derecho de instalar un programa en un ordenador "personal". Por ejemplo, la licencia de uso de un procesador de textos o de un gestor de hojas de cálculo.

En ocasiones, la compra de un software personal incluye, en la licencia, la posibilidad de instalarlo en varios ordenadores personales (2-3). Esta política es muy seguida por los antivirus, de manera que al comprar uno lo puedes instalar en tu ordenador personal, en el portátil y en el ordenador de la habitación del hijo, por ejemplo.

En cambio, una licencia servidora corresponde al derecho de instalar y poder utilizar un programa que atenderá "peticiones" de otros programas "clientes". Ejemplos de licencias servidoras son las correspondientes a los servidores de bases de datos, servidores de correo, servidor de acceso a Internet, etc.

Finalmente, a una licencia "cliente" corresponde el derecho de que un ordenador acceda y trabaje con un programa que se ha instalado como servidor. De este modo, para cada ordenador o usuario que acceda al servidor de bases de datos o al servidor de correo, se debe adquirir una licencia cliente correspondiente.

Los clientes potenciales de las licencias servidoras y clientes son empresas y grandes corporaciones. La obtención de licencias servidoras y clientes van muy de la mano; a menudo, se compra una o dos licencias servidoras y tantas licencias clientes como ordenadores esté previsto que accedan a los datos del servidor. Por ejemplo, si se tratara de un programa gestor de correo electrónico, se tendría que comprar una licencia servidora y tantas licencias cliente como cuentas de correo se quieran tener.

El uso ilegal de un programa supone un delito y, como tal, está penalizado.

Hay varias organizaciones, nacionales e internacionales, encargadas de perseguir la piratería informática en las compañías y los organismos.

Las licencias se pueden clasificar de manera simplificada en personal, de servidor y de cliente.

4.1. Políticas de compra y uso de licencias

Cada fabricante tiene una política de venta de licencias diferente y particular.

Estas políticas, además, evolucionan con el tiempo y se adaptan a las nuevas características de las tecnologías de la información (TI).

Así, la aparición de aplicaciones Internet supuso la aparición no sólo de un canal nuevo de venta y distribución, sino también de nuevas políticas de compra y uso de licencias como, por ejemplo, las aplicaciones Web.

Aplicaciones web

Una aplicación web está instalada en un proveedor de Internet, de manera que el cliente no es propietario de esta aplicación, sino que alquila su uso para llevar a cabo algunos procesos que requieren esta solución, de manera análoga al alquiler de un coche para hacer un viaje.

Algunos ejemplos son los siguientes:

- Microsoft office en línea: se pueden crear, editar y guardar documentos de MS Office (Excel, Word, Access, PowerPoint, etc.) sin tener instalado el MS Office. Para hacerlo, hay que registrarse en la web que MS tiene dedicada a este servicio.
- Programas convertidores de archivos: se pueden encontrar en la Red sitios web dedicados a la conversión de archivos libres de tasas, por ejemplo, para convertir documentos de Word o imágenes en pdf.
- Antivirus en línea: la mayoría de empresas del sector tienen una parte de su web dedicada a la detección de virus en línea. De esta manera, desde allí, y sin necesidad de comprar una licencia de antivirus, hay la posibilidad de escanear y detectar virus en vivo. La mayoría de estos servicios, sin embargo, sólo te informan de la existencia o no del virus en tu PC; para eliminarlo, hay que comprar la licencia.

Una vez decidida la necesidad de adquirir un programa, es necesario seguir tres pasos básicos para comprar las licencias de software:

a) Determinar la licencia necesaria (tipo de software y cantidad).

b) Decidir la mejor opción entre comprar o *leasing*.

c) Determinar dónde realizar la compra.

Para determinar la licencia necesaria debemos tener en cuenta los datos o categorías siguientes:

- **Tipo de producto**. Algunos fabricantes clasifican sus programas según estos tres tipos:
  - **Aplicaciones**. Programas genéricos de productividad personal, utilidades o herramientas de desarrollo.
  - **Sistemas**. Corresponden a los sistemas operativos.
  - **Servidores**. Enmarca todas las aplicaciones corporativas o servidores corporativos, como por ejemplo, los servidores de bases de datos, de Internet o de correo.
- **El producto**. El programa en sí, según las funcionalidades requeridas. Por ejemplo, un procesador de textos, un antivirus, etc.
- **Versión**. Un mismo producto puede contar con diferentes versiones, tanto por la evolución de los sistemas operativos en los que se pueden instalar y ejecutar, como por disponer de más o menos funcionalidades y prestaciones.
- **Edición**. Especifica el conjunto de prestaciones y/o aplicaciones incluidas en el producto.
  Un producto puede disponer, por ejemplo, de las ediciones estándar, profesional y experto.
- **Tipo de licencia.** En función del fabricante pueden existir diferentes tipos como, por ejemplo, la licencia de nuevo usuario, correspondiente a la compra inicial del producto, la actualización de versión, que permite la adquisición de una versión posterior del programa si se dispone de la licencia de la versión anterior del mismo producto, o la actualización competitiva, que corresponde a la compra de una licencia de "coste menor" de un programa, si se dispone de una licencia de un producto relacionado de otro fabricante.

Aun así, estos son ejemplos genéricos de tipos de licencias, ya que cada fabricante posee su política específica de compra y uso de sus programas.

Algunos fabricantes tienen una política de venta relacionada con el uso de la herramienta por parte del cliente, cobrando, por ejemplo, una comisión por cada transacción realizada (plataformas de eCommerce) utilizando la aplicación. Este último tipo de uso de licencias suele corresponder a sistemas de comercio electrónico donde el comprador abona un importe base (coste fijo) para la herramienta y una comisión (coste variable) para cada transacción que se realiza mediante ésta.

Hay opciones muy económicas que permiten a las empresas disfrutar de software de alta calidad sin necesidad de comprar el software ni el hardware. Así sólo se paga por la licencia; tanto el software como el hardware están físicamente en casa del fabricante; lo que se hace es reservar una parte de los recursos del hardware y configurar el software según las necesidades del cliente. Suelen ser aplicaciones web, de manera que se puede acceder con un navegador desde la empresa y hace el mismo uso como si fuera una aplicación instalada en tu propio hardware. Un ejemplo de este caso es el Siebel, una herramienta CRM muy potente y con un coste económico muy grande que, en su versión Siebel *on demand*, funciona como se ha descrito y cualquier pyme puede acceder a Siebel, una herramienta CRM, hasta ahora reservada a corporaciones con un gran volumen de negocio.

Hay diferentes opciones de compra del software:

- **OEM (*original equipment manufacturas*).** Corresponde al software incluido en la compra de un ordenador o un equipo informático nuevos. El caso más común es cuando, al comprar un ordenador personal, el fabricante incluye la licencia del sistema operativo y un conjunto de aplicaciones de productividad personal: procesador de textos, hoja de cálculo, herramienta de dibujo, etc.
  El software OEM es una versión especial del software que se debe distribuir de manera preinstalada en el disco duro del PC.
  Nunca se puede distribuir software por OEM sin el PC o el hardware correspondiente.

- **Compra del paquete.** Corresponde a la compra física de la caja con el producto en cualquier tienda o almacén.
  Incluye el producto en formato magnético, por ejemplo, CD o DVD junto a las licencias correspondientes.

- **Licencia por volumen para organizaciones.** Este tipo de compra corresponde a las realizadas por las compañías que necesitan una gran cantidad de licencias del mismo software.

Ejemplos

Por ejemplo, una organización de 200 trabajadores puede necesitar, por ejemplo, adquirir 110 licencias de un procesador de textos. En estos casos, en lugar de adquirir los 110 CD o DVD del producto, se compran dos o tres CD y las 110 licencias (el derecho de instalar el producto en 110 máquinas). Esto reduce el coste del producto y los costes de

envío. La mayoría de los fabricantes aplican diferentes políticas de precio dependiendo del número de licencias adquiridas o del tipo de organización (administración pública, instituciones académicas, etc.).

Por ejemplo, Microsoft distingue entre la venta a pequeñas y medianas empresas (pymes) con la Open Multilicencia (adquisición de licencia perpetua) o la Open Suscripción (licencias no perpetuas) y la venta a grandes empresas, con el Contrato Select o Enterprise Agreement Suscription.

Una vez decidida la necesidad de adquirir un programa, se debe determinar la licencia necesaria (tipo de software y cantidad), decidir la mejor opción entre comprar o *leasing* y determinar dónde se debe realizar la compra.

4.2. Licencias de software gratuito (*freeware*) y software de prueba (*shareware*)

En los apartados anteriores se ha introducido el concepto de software y sus aplicaciones.

Una solución informática es un "programa" que se instala en un ordenador o hardware para llevar a cabo ciertas tareas, ya sea para elaborar y editar documentos, gestionar bases de datos, ofrecer conectividad entre ordenadores o para gestionar proyectos.

Utilizar estos programas implica tener una o más licencias.

Se han comentado diferentes políticas de compra y uso de licencias de software.

Ahora bien, hay dos "tipos de licencias" no consideradas en los apartados anteriores y que cada vez tienen más importancia y resultan más usuales: las licencias de software gratuito (*freeware*) y las licencias de software de prueba (*shareware*).

Según la disponibilidad de los archivos fuente, el software se puede clasificar en abierto (libre, de dominio público y semilibre) y cerrado (gratuito, de prueba, de demostración y propietario).

Según el coste que representa para el usuario, el software se puede clasificar en gratuito (libre, de dominio público, semilibre y gratuito) y en software no gratuito (de prueba, de demostración y propietario).

- El software **gratuito** se puede utilizar, copiar y distribuir libremente, pero no modificar, ya que no incluye el código fuente. Para la FSF (Free Software Foundation), entidad que promueve el uso y desarrollo de software de este tipo, el software gratuito no es libre, aunque tampoco lo califica como

semilibre ni como propietario. Es un software que no es necesario comprar para utilizarlo.

- El software **de prueba** se caracteriza por que es de libre distribución o copia, de manera que se puede utilizar, si se cuenta con el permiso del autor, durante un período limitado de tiempo, y se debe pagar, transcurrido éste, para continuar utilizándolo, aunque la obligación es sólo de tipo moral, ya que los autores entregan los programas confiando en la honestidad de los usuarios. Este software se suele distribuir sin el código fuente y no se permite modificarlo ni redistribuirlo. Muchas veces, por ignorancia, los programas de esta clase se suelen utilizar de manera ilegal. A menudo se confunde el software de prueba con el software de demostración, que son programas no funcionales al cien por cien o que dejan de funcionar después de un tiempo determinado.

Según la disponibilidad de los archivos fuente, el software se puede clasificar en abierto (libre, de dominio público y semilibre) y cerrado (gratuito, de prueba, de demostración y propietario); y según el coste que representa para el usuario, se puede clasificar en gratuito (libre, de dominio público, semilibre y gratuito) y en software no gratuito (de prueba, de demostración y propietario).

Hay casos cuyo régimen está a caballo entre los mencionados, en los que se permite el uso del software hasta el momento en que se usa por tareas lucrativas; entonces, se tiene que pagar la licencia. Es decir, son *freeware* hasta el momento en que se usa para sacar un rendimiento económico, y luego hay que hacer el pago de la licencia. Por ejemplo, la base de datos Oracle.

No debemos confundir, sin embargo, las licencias de software gratuito con el código fuente abierto u *open source*.

Ejemplos de aplicación gratuita

Ejemplos de aplicaciones gratuitas son los siguientes:

- Paquetes ofimáticos, con procesador de textos, hojas de cálculo, etc.: OpenOffice.org, Star Office y los gestores de bases de datos DBMaker 4.03 y TreeDBNotes Free.
- Sistemas operativos: OPTIX IV, BeOs y Linux MandrakeSoft.
- Gestores de correo electrónico: FoxMail 4, Desktop Sidebar y Cactus Mail.
- Programas de retoque fotográfico: PhotoPlus y Photo Librarian Image Editor.
- Programas para realizar copias de seguridad: WinDriversBackup y InFoAL BackUp.

Open source

Nombre que se da al software que se puede distribuir libremente con la única condición de que siempre se dé el código fuente junto al ejecutable. En caso de que el código contenga errores de programación, el usuario los puede corregir. Según cuál sea la licencia de código fuente abierto, el usuario también tiene el permiso de distribuir la versión del programa que haya modificado. Los sistemas operativos Linux o la *suite* ofimática OpenOffice son el paradigma del software *open source*.

Ejemplos de aplicación de prueba

Ejemplos de aplicaciones de prueba son los siguientes:

- Gestores de bases de datos: SQLConnect 1.6.2.
- Programas de diseño gráfico: 3D Flash Animator, Paint Shop Pro e IntelliCAD 2001 Standard.
- Editores de páginas web: Ace Expert, CoffeeCup HTML Editor y Web Design.
- Antivirus: Anti-Trojan y MC Afee VirusScan 14.4.0.
- Gestor de correo electrónico: Calypso.
- Programas de gestión de copias de seguridad: mCopias, Magellan y Backup BaK Again II Workstation.

4.3. La piratería de software

El término *piratería de software* cubre diferentes actividades, desde copiar programas ilegalmente, hasta falsificar y distribuir software sin el consentimiento o acuerdo con el fabricante correspondiente.

4.3.1. Consecuencias de la piratería de software

La mayoría de las personas creen que la piratería de software sólo afecta a la industria informática.

En cambio, se ha de tener en cuenta que, actualmente (12/05/2009), el 42% del software que se utiliza en España está copiado ilegalmente. A escala mundial, el porcentaje es del 41%. Con estos datos, se puede dimensionar el alcance real del problema.

Según la BSA (Bussiness Software Alliance), la piratería informática causa estragos no sólo en la industria del sector, sino también en la sociedad; a los millones de euros en pérdidas del sector hay que añadir los millares de puestos de trabajo que se pierden o se dejan de generar debido a la piratería. Se pueden encontrar muchos informes de actualidad referidos a la piratería informática, desde el punto de vista de de la industria del software en la web de la BSA (indicada arriba).

Un estudio sobre piratería de software realizado por International Planning & Research Corp., también durante el año 2000, señala que estas acciones ilegales representaron la pérdida de 118.026 puestos de trabajo en Estados Unidos.

La utilización ilegal del programa también supone otros gastos y costes a las organizaciones que lo utilizan.

Ejemplo de costes derivados

Ejemplos de estos costes derivados son los siguientes:

- la falta de apoyo técnico, por parte del fabricante,
- el acceso a nuevas versiones, actualizaciones, adaptaciones y resolución de errores (garantía posventa),
- el acceso a programas completos con todas las funcionalidades,
- disponer de programas libres de virus.

Asimismo, la violación del Código Penal expone a las compañías a varias acciones judiciales y al desprestigio y la pérdida de su imagen.

Resumiendo, se podrían clasificar estas consecuencias según tres tipos de destinatarios: los consumidores, los desarrolladores y los vendedores.

Cuando un usuario decide hacer una copia no autorizada de un programa, está renunciando al derecho a la asistencia, documentación, garantías y actualizaciones periódicas. Si la copia ilegal del software se obtiene por medio de programas P2P, el riesgo de que el fichero esté infectado por algún virus es muy alto.

P2P o *peer 2 peer*

Son programas que ponen en contacto dos ordenadores remotos y se pueden compartir ficheros. Son programas P2P el Emule o el BitTorrent, por citar dos de los más populares.

Al piratear un producto protegido por las leyes de propiedad intelectual, el individuo se expone al riesgo legal que esto conlleva.

La pérdida de ingresos que implica la piratería se habría podido invertir en el producto y conseguir, así, reducir el precio final para el consumidor. Con estas acciones ilegales se perjudican muchas y variadas empresas que tienen su medio de subsistencia en el diseño y desarrollo de software y en su distribución y venta.

La piratería de software no sólo afecta a la industria informática, sino también a los consumidores, desarrolladores y vendedores.

4.3.2. Formas de piratería

Habitualmente, cuando se piensa y se habla de piratería de software se identifica con las copias. Aunque ésta es una de las formas más extendidas de piratería, hay muchas otras:

- **Copias realizadas por el usuario final**. Son simples copias sin licencia realizadas por usuarios individuales o empresas. También se pueden considerar, en el caso de la compra de licencias por volumen, una información incorrecta sobre el número de ordenadores en los cuales se ha instalado el software.

- **Carga en el disco duro**. Tal como hemos comentado en el apartado anterior, a menudo los fabricantes de ordenadores tienen contratas con los fabricantes de software que les permiten distribuir software por OEM (soft-

ware que se debe distribuir de manera preinstalada en el disco duro del PC). Un tipo muy habitual de piratería lo llevan a cabo fabricantes de sistemas de PC que comercializan los ordenadores con software preinstalado sin que este software sea legal. Es decir, sin haber firmado un contrato con el fabricante del software para distribuirlo.

- **Falsificaciones**. Es la piratería de software a gran escala, en la que se duplica ilegalmente el software y sus cajas, llevado a cabo por redes organizadas que distribuyen los productos como supuestamente legales.

- **Distribución por canales fraudulentos**. Muchos fabricantes de software distribuyen licencias a colectivos determinados como, por ejemplo, el colectivo educativo, con unos descuentos especiales. Otras veces, por compras de gran volumen, los fabricantes ofrecen unos descuentos al comprador (grandes empresas, administraciones públicas). Una casuística de piratería corresponde al software distribuido como licencias con un descuento especial, a clientes con un gran volumen, a fabricantes de ordenadores o a entidades académicas, que se redistribuye, posteriormente, a otros usuarios que no cumplen los requisitos para disponer de estas licencias.

- **Piratería en Internet**. En este caso, corresponde más al medio que al tipo. Se utiliza Internet para la distribución ilegal del software, de manera que se puede acceder a copias de los diferentes programas sin adquirirlos al fabricante o a los distribuidores autorizados. Algunos informes de la BSA hablan de que dos de los grandes canales de distribución de falsificaciones y copias ilegales de software son los programas P2P y el portal de subastas Ebay. El tráfico de datos en Internet debido a los programas P2P es tan alto que, según fuentes de la BSA, en algunos países llega a situarse entre el 49%-89% del tráfico total diurno en Internet, y la cifra se eleva hasta el 95% de noche.

Otros ejemplos

Otras modos de piratería pueden ser, por ejemplo, instalar un producto original en varios ordenadores sin haber adquirido el número de licencias necesario o instalar en el ordenador personal un software legal pero adquirido para utilizarlo en el entorno laboral.

La piratería de software va más allá del simple hecho de copiarlo ilegalmente. Hay otras maneras de piratear como la carga ilegal en el disco duro, las falsificaciones o el uso de más licencias que las adquiridas legalmente.

Ejercicios de autoevaluación

1. Clasificad los programas y las aplicaciones siguientes siguiendo la taxonomía presentada en el apartado "Aplicaciones del software".

Aplicación para el control de los pedidos de los clientes de una empresa.

- Simulador de una constelación y/o del movimiento de los planetas del sistema solar.
- Sistema de gestión de proyectos.
- Programa de reconocimiento de la voz.
- Sistema operativo.
- Programa de interfaz de comunicaciones entre un ordenador y un dispositivo móvil.
- Herramienta de apoyo del control aéreo.
- Sistema de visión artificial.
- Aplicación de agenda.
- Herramienta de apoyo al diseño de piezas.
- Geográfico y seguimiento de un paquete.
- Analizador de los componentes de una solución química.
- Sistema de control de los semáforos de una ciudad.
- Herramienta de posicionamiento.

2. Ordenad, cronológicamente, los lenguajes presentados en el apartado "Principales lenguajes de desarrollo": Pascal, Eiffel, Cobol, Simula, C, Fortran, Basic, C++, Lisp, Algol, RPG, Modula2, Ada, Prolog.

3. Clasificad los lenguajes presentados en las cuatro categorías siguientes:

Lenguajes de aplicación en inteligencia artificial	Lenguajes de aplicaciones científicas	Lenguajes orientados a objetos	Lenguajes de propósito general	Generadores de programas

(Si creéis que un lenguaje puede estar en más de una categoría, indicadlo).

4. Clasificad los elementos de ingeniería del software siguientes según si son un método, una herramienta o un procedimiento:

- Redacción y generación de informes.
- Etapas del ciclo de vida.
- Proceso de verificación de una solución.
- Generador del modelo de bases de datos.
- Sistema de gestión de proyectos.
- Programación modular y programación estructurada.
- Diseño del modelo conceptual de una base de datos y generación del modelo físico.
- Generador de formularios de entrada de datos.

5. Vistas las tareas siguientes, indicad cuáles creéis que podrían ser hitos:

- Presentación informe detallado del análisis de requerimientos.
- Inicio del proyecto.
- Redacción y generación de informes.
- Diseño del modelo conceptual de la base de datos.
- Fichero generado.
- Generación del modelo físico de la base de datos.
- Entrega del modelo conceptual de la base de datos.
- Módulo validado por el cliente.
- Inicio fase mantenimiento.
- Copias de seguridad validadas.

6. Indicad para cada una de las interrelaciones entre tareas si creéis que son del tipo fin-inicio, fin-fin, inicio-inicio o inicio-fin:

- Lectura secuencial de un fichero de datos-generación de la media de los valores leídos.
- Validación diseño detallado-desarrollo de los módulos de software.
- Traslado material del almacén del proveedor a casa del cliente-vigilancia del material del almacén del proveedor.

- Traducción del texto de un idioma a otro-revisión ortográfica y gramatical de un texto.

7. A partir de la siguiente descripción del trabajo pendiente, identificad las posibles tareas, sus interrelaciones y dibujad el diagrama de Gantt.

"El objetivo del proyecto es la producción y distribución de una nueva línea de perfumes de señor para que se presente en la Feria Internacional del Perfume de París del próximo año".

Para poder llevar a cabo este proyecto, se deben realizar una serie de tareas:

- Realizar un estudio de mercado (4 semanas).
- Desarrollar el diseño del producto y las especificaciones de coste (2 semanas + 1 semana).
- Producir un prototipo (3 semanas).
- Probarlo (1 semana).
- Coordinar la producción con el equipo de fabricación (8 semanas).
- Diseñar y producir el envase (2 semanas + 4 semanas).
- Envasado del producto (1 semana).
- Coordinar las campañas de publicidad para su lanzamiento al mercado (4 semanas).

Y con el fin de controlar el avance del proyecto, se ha decidido establecer los puntos de control siguientes:

- Análisis de mercado completo.
- Diseño y precio aprobados.
- Prototipo finalizado para las pruebas.
- Prototipo aprobado para la producción.
- Producto empaquetado y enviado al almacén.
- Producto a punto para salir al mercado.

8. A partir del ejercicio anterior, identificad el camino crítico del proyecto, indicando qué tareas forman parte de este camino crítico y, de manera intuitiva, indicad cuál sería el margen de cada una de las tareas.

9. Asociad para cada una de las tareas indicadas qué rol del equipo de trabajo de los mencionados las asumiría.

Tareas:

- Instalación de los sistemas operativos de las máquinas y su configuración.
- Diseño de los iconos del programa.
- Coordinación del equipo de ingeniería.
- Definición de los productos y la tipología para una solución de ofertas comerciales.
- Diseño del plan de comunicación del lanzamiento de una nueva web en el mercado.
- Gestión y control de las horas y del presupuesto del proyecto.
- Análisis del proceso de compraventa en un mercado virtual.
- Programación de las rutinas de código del programa.
- Instalación y configuración del servidor de bases de datos.
- Comprobación del cumplimiento de los requerimientos del diseño gráfico de la solución.
- Coordinación del equipo de aseguramiento de la calidad.

Roles:

- Consultor de procesos.
- Consultor de catálogos.
- Consultor de marketing.
- Diseñador gráfico.
- Director técnico.
- Ingeniero de software.
- Administrador de bases de datos.
- Administrador de sistemas.
- Verificador de "Look & Feel".
- Responsable de calidad.
- Administrador del proyecto.

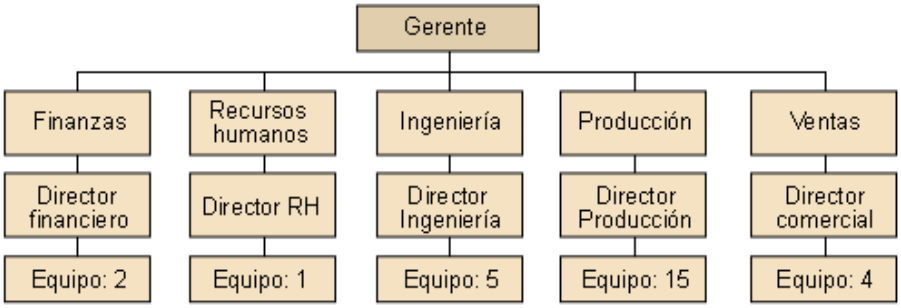
10. A continuación indicamos la estructura organizativa de una empresa y el personal y los cargos de cada departamento.

Se ha decidido utilizar el sistema operativo SOP1, que requiere licencia servidor y una licencia cliente para cada usuario que acceda a él, y el procesador de textos PTX1, que requiere una licencia cliente para cada usuario que lo utilice. Lo mismo sucede con el programa de gestión

de hoja de cálculo FCAL1, el sistema de gestión de proyectos SGP1 y el programa de antivirus AVS1.

Así, se adquirirá el sistema de gestión de mensajería electrónica, SME1, que implica los mismos requerimientos de licencias que el sistema operativo SOP1.

Se pide identificar las licencias de servidor y cliente de cada aplicación, teniendo en cuenta la estructura organizativa indicada y las necesidades funcionales de la compañía siguientes:



- Se dispondrá de un único servidor de ficheros e impresoras (sistema operativo) en el que también se instalará el servidor de mensajería electrónica.
- Todos los trabajadores tienen acceso a este servidor y al correo electrónico.
- Sólo el equipo de ingeniería, el director de producción y el gerente han de gestionar proyectos.
- Los responsables de cada departamento, el gerente y los miembros de los departamentos de Finanzas y RRHH deben trabajar con el procesador de textos y la hoja de cálculo.
- Todo el personal debe tener instalado en la máquina el programa antivirus.

Solucionario

Ejercicios de autoevaluación

1. Aplicación para el control de los pedidos de los clientes de una empresa.

- Software de sistemas
  Sistema operativo. Programa de interfaz de comunicaciones entre un ordenador y un dispositivo móvil.
- Software de tiempo real
  Herramienta de apoyo al control aéreo. Sistema de control de los semáforos de una ciudad. Herramienta de posicionamiento geográfico y seguimiento de un paquete.
- Herramientas de uso personal
  Aplicación de agenda. Herramienta de apoyo al diseño de piezas.
- Software de ingeniería y científico
  Analizador de los componentes de una solución química. Simulador de una constelación y/o del movimiento de los planetas del sistema solar.
- Herramientas de gestión y rediseño de procesos organizativos
  Sistema de gestión de proyectos. Aplicación para el control de los pedidos de los clientes de una empresa.
- Aplicaciones de inteligencia artificial
  Programa de reconocimiento de la voz. Sistema de visión artificial.

2.

Fortran (1957)
Lisp (1959)
Cobol (1960)
Simula (1962)
RPG (década los sesenta)
Algol (década de los sesenta)
Basic (1965)
Pascal (1970)
C (1972)
Prolog (década de los setenta)
Modula2 (finales de los setenta)
C++ (década de los ochenta)
Ada (1983)
Eiffel (1986)

3.

Lenguajes de aplicación en inteligencia artificial	Lenguajes de aplicaciones científicas	Lenguajes orientados a objetos	Lenguajes de propósito general	Generadores de programas
Prolog, Lisp, "Simula"	Fortran, "Algol", PL1	C++, Simula, Smalltalk, Eiffel	Cobol, Algol, PL1, Basic, C, Pascal, Modula2, Ada	RPG

4.

Método	Etapas del ciclo de vida, programación modular y programación estructurada, diseño del modelo conceptual de una base de datos y generación del modelo físico.
Herramienta	Generador del modelo de bases de datos, sistema de gestión de proyectos, generador de formularios de entrada de datos.
Procedimiento	Redacción y generación de informes, proceso de verificación de una solución.

5. Aunque dependerá de cada proyecto, del significado de la tarea y del esfuerzo necesario para llevarla a cabo, en aquel proyecto se podrían considerar hitos, de manera general, las tareas siguientes:

- Entrega del modelo conceptual de la base de datos.
- Módulo validado por el cliente.
- Presentación informe detallado del análisis de requerimientos.
- Inicio del proyecto.
- Inicio de la fase mantenimiento.
- Fichero generado.
- Copias de seguridad validadas.

Todas indican acciones completadas o "puntos de control" dentro del proyecto.

En cambio, las tareas:

- redacción y generación de informes,
- diseño del modelo conceptual de la base de datos,
- y generación del modelo físico de la base de datos

representan una "tarea" que se debe realizar con una duración para completarla.

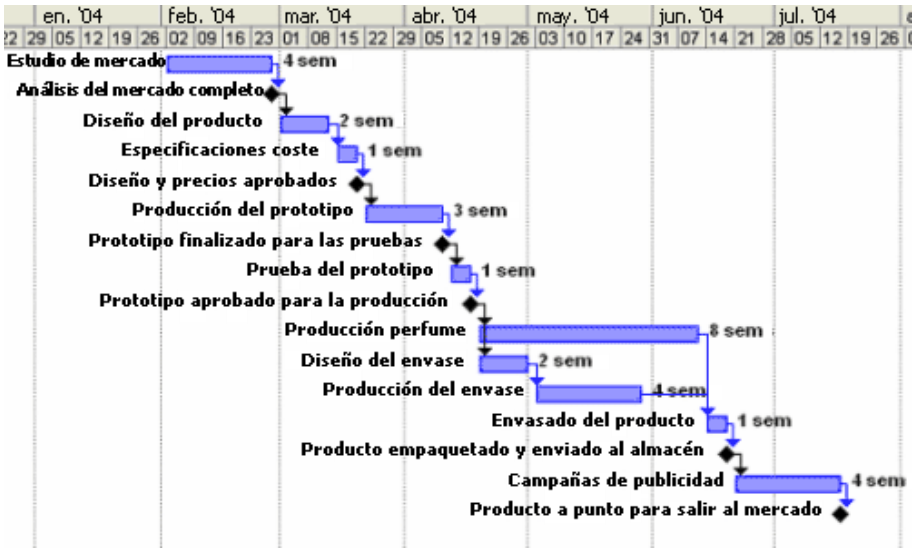
6.

- Validación diseño detallado-desarrollo de los módulos de software (fin-inicio).
  Dado que hasta que no se ha validado el diseño detallado de las funcionalidades, no se pueden empezar a desarrollar los módulos según este diseño.
- Traducción del texto de un idioma a otro-revisión ortográfica y gramatical de un texto (inicio-inicio).
  Se pueden realizar las dos tareas en paralelo, de manera que cuando empiece la traducción, también pueda empezar la revisión.
  En este caso, también se podrían secuenciar las tareas, de manera que cuando acabe la traducción, se inicie la corrección.
- Lectura secuencial de un fichero de datos-generación de la media de los valores leídos (fin-fin).
  Considerando que a medida que se va leyendo el fichero de datos, se puede ir calculando la media de los valores leídos hasta el momento, habría una relación fin-fin entre las dos tareas. También se podría considerar una relación fin-inicio, si la media no se calculara hasta el final de la lectura de todo el fichero, en lugar de calcularse después de leer cada valor.
- Traslado material del almacén del proveedor a casa del cliente-vigilancia del material del almacén del proveedor (inicio-fin).
  En el momento en el que se inicia el traslado del material del almacén del proveedor al cliente, se detiene la tarea de vigilancia del material que hay en el almacén (inicio-fin).

7. Se descompone el proyecto en tareas, se intenta poder efectuar tareas en paralelo, con el fin de adelantar el trabajo (producción del perfume y diseño y producción del envase):

NÚM: TAREA	TAREA	DURACIÓN	PREDECESORA
1	Estudio de mercado	4 sem.	
2	Análisis del mercado completo	0 días	1
3	Diseño del producto	2 sem.	2
4	Especificaciones coste	1 sem.	3
5	Diseño y precios aprobados	0 días	4
6	Producción del prototipo	3 sem.	5
7	Prototipo finalizado para las pruebas	0 días	6
8	Prueba del prototipo	1 sem.	7
9	Prototipo aprobado para la producción	0 días	8
10	Producción perfume	8 sem.	9
11	Diseño del envàse	2 días	9
12	Producción del ehvase	4 sem.	11
13	Envasado del producto	1 sem.	12;10
14	Producto empaquetado y enviado al almacén	0 días	13
15	Campañas de publicidad	4 sem.	14
16	Producto a punto para salir al mercado	0 días	15

El diagrama de Gantt tendría el aspecto siguiente:



8. El camino crítico del proyecto está compuesto por las tareas que, si se retrasan, provocarán, seguro, el atraso de la fecha de fin del proyecto.

En el ejemplo, son todas las tareas excepto las tareas de diseño del envase y de producción de éste.

Para evaluar los márgenes de las diferentes tareas, es necesario centrarse en las tareas que no forman parte del camino crítico.

Las tareas críticas no tienen margen, ya que si alguna se retrasara, el proyecto también se retrasa.

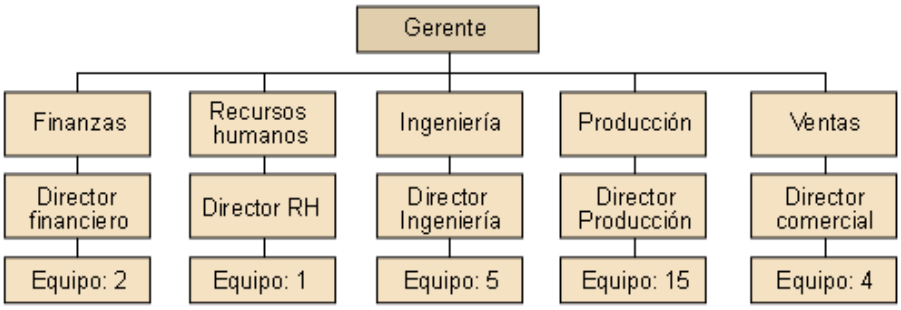
En cambio, las tareas de diseño del envase y de producción del envase tienen un margen de atraso posible.

De manera intuitiva, estas dos tareas (duración prevista 2 + 4 semanas) se realizan en paralelo con la producción del perfume (con una duración prevista de ocho semanas), por lo que tienen un margen posible de dos semanas de atraso. Si se retrasan más de dos semanas, pueden afectar al fin del proyecto.

9.

Roles	Tareas
Consultor de procesos	Análisis del proceso de compraventa en un mercado virtual
Consultor de catálogos	Definición de los productos y tipología para una solución de ofertas comerciales
Consultor de marketing	Diseño del plan de comunicación del lanzamiento de una web nueva al mercado
Diseñador gráfico	Diseño de los iconos del programa
Director técnico	Coordinación del equipo de ingeniería
Ingeniero de software	Programación de las rutinas de código del programa
Administrador de bases de datos	Instalación y configuración del servidor de bases de datos
Administrador de sistemas	Instalación de los sistemas operativos de las máquinas y su configuración
Verificador de "Look & Feel"	Comprobación del cumplimiento de los requerimientos del diseño gráfico de la solución
Responsable de calidad	Coordinación del equipo de aseguramiento de la calidad
Administrador del proyecto	Gestión y control de las horas y del presupuesto del proyecto

10.



Total trabajadores compañía y por departamento:

- Gerencia: 1
- Finanzas: 1 + 2
- Recursos Humanos: 1 + 1
- Ingeniería: 1 + 5
- Producción: 1 + 15
- Ventas: 1 + 4
- Total empresa: 33

Licencias requeridas:

- Sistema operativo (SOP1): 1 licencia servidor + 33 licencias cliente
- Sistema mensajería electrónica (SME1): 1 licencia servidor + 33 licencias cliente
- Sistema de gestión de proyecto (SGP1): 8 licencias
- Procesador de textos (PTX1): 9 licencias
- Hoja de cálculo (FCAL1): 9 licencias
- Antivirus (AVS1): 33 licencias

Bibliografía

Barceló, M.; Costa, M.; Quer, C. (1993). *Anàlisi d'aplicacions informàtiques*. Barcelona: Edicions UPC.

Pressman, Roger S. (1997). *Ingeniería de Software. Un enfoque práctico*. Madrid: McGraw-Hill.

Drudis, A. (1999). *Gestión de Proyectos. Cómo planificarlos, organizarlos y dirigirlos*. Barcelona: Ediciones Gestión 2000.

Villarreal, S. (1999). *Introducción a la computación*. México DF: McGraw-Hill.

Domingo Ajenjo, A. (2000). *Dirección y gestión de proyectos*. Madrid: Editorial Ra-Ma.

Redes y comunicaciones

Jeroni Vivó i Plans

PID_00153115

Universitat Oberta de Catalunya
www.uoc.edu

Índice

Objetivos

Los **objetivos generales** que el estudiante puede alcanzar son los siguientes:

1. Conocer el esquema general de un sistema de telecomunicación.
2. Conocer el concepto de red, sus tipologías y funcionamiento, y sus implicaciones prácticas.
3. Conocer los fundamentos técnicos de la red Internet y de las aplicaciones web.
4. Identificar los rasgos fundamentales de un sistema de telecomunicación o una red, ante situaciones prácticas de contratación de servicios o adquisición de productos relacionados.

Estos objetivos generales se desglosan en los **objetivos específicos** siguientes:

1. Entender el concepto de sistema de telecomunicación y sus elementos.
2. Entender el proceso de digitalización y su conveniencia.
3. Entender la amenaza que representa el ruido para la telecomunicación y la necesidad del procesamiento de señal.
4. Ejemplarizar el funcionamiento de un sistema de telecomunicación.
5. Definir red de telecomunicación o informática.
6. Entender al modelo OSI de capas de funcionamiento de una red.
7. Clasificar las redes según extensión, topología física.
8. Entender los protocolos de funcionamiento de la red.
9. Introducir las redes sin hilos.
10. Definir la red Internet y conocer sus orígenes.
11. Conocer el protocolo de comunicación en Internet: TCP/IP.

12. Identificar servicios construidos sobre la red Internet y los respectivos protocolos.

13. Definir una aplicación web y describir su arquitectura básica.

14. Definir un servicio web y describir su arquitectura básica.

15. Definir intranet y extranet.

16. Entender la importancia del ancho de banda para el dimensionado adecuado de un sistema de telecomunicación.

17. Conocer a grandes rasgos los servicios comerciales de transportes de datos.

1. Sistema de telecomunicación

Aunque las telecomunicaciones son una ciencia mucho más antigua que la informática, desde su aparición las dos han ido de la mano. Todo sistema de telecomunicaciones posee un componente muy importante de software y a la vez todo ordenador necesita comunicarse con otros ordenadores para poder extraer todo su potencial.

Desde el punto de vista de la gestión de información, nos interesa conocer descriptivamente cómo funciona un sistema de telecomunicación para poder actuar con solvencia como responsable de un proyecto ante la necesidad de comunicar fuentes de información.

Si estas fuentes no están en soporte informático, se deben digitalizar, por lo que también es necesario conocer el proceso de la digitalización.

A lo largo de la unidad se realiza un recorrido por los elementos que integran una comunicación y se extrapola a un sistema de telecomunicación.

Se explica la diferencia entre un sistema analógico y uno digital y se evalúa cuál de los dos es mejor y por qué. Para hacerlo se parte de un sistema analógico y se ve cómo se puede convertir en digital, se explica cada paso del proceso hasta llegar a una señal puramente digital. El objetivo no es conocer al detalle el proceso de digitalización de una señal, sino entender la filosofía, el concepto de sistema digital y ver que las diferencias entre un sistema digital y uno analógico los hace incompatibles.

Se detallará un proceso de digitalización para entender bien qué hay tras la palabra *digital*. Es importante extraer el concepto, ya que aquí se detalla la digitalización de una señal pero el concepto es igual para una imagen, un documento de texto, etc.

Para evaluar la calidad de los dos sistemas, digital y analógico, se habla de ruido. El ruido delata las virtudes y los defectos de cada uno de los sistemas y podremos sacar nuestras propias conclusiones.

Finalmente, y a modo de resumen, aparece un ejemplo de un sistema de telecomunicación conocido por todos, la radio.

1.1. Comunicación y sistema de telecomunicación

Uno de los rasgos distintivos de la raza humana es su especial habilidad para comunicarse. A lo largo de la historia, esta habilidad se ha ido desarrollando para dar como fruto una rica variedad de formas de comunicación: la escritura, la comunicación gestual, la comunicación verbal con todas las diferentes lenguas y dialectos, etc.

Por muchos y variados motivos, el hombre tiene la necesidad de introducir elementos nuevos en la comunicación en situaciones en las que los interlocutores no se encuentran cara a cara, que sería la forma natural de la comunicación. Para poder seguir disfrutando de los beneficios de la comunicación sin necesidad de tener una interacción directa entre los diferentes interlocutores y superar, así, la no coincidencia en el espacio y/o tiempo de los interlocutores, aparecen los primeros sistemas de telecomunicación.

Los sistemas de telecomunicación son una evolución de lo que entendemos propiamente como comunicación. Veamos cómo definen comunicación algunos de los organismos más distinguidos.

La Real Academia de la Lengua Española define *comunicación* en una de sus múltiples acepciones como:

> "Transmisión de señales mediante un código común al emisor y al receptor".

El *Diccionari general de la llengua catalana* de Pompeu Fabra nos dice que por *comunicación* se entiende:

> "Proceso de interacción que se produce entre un emisor y un receptor, seres o sistema, que consiste en el paso de información del primero al segundo mediante un código que se transmite por medio de un canal".

La *Encyclopedia Britannica* nos da como definición de comunicación (*communication*):

> "The exchange of meanings between individuals through a common system of symbols" ('intercambio de significados entre individuos por medio de un sistema de símbolos conocido por ambos')".

Las mismas fuentes nos definen *telecomunicación* como:

> "Sistema de comunicación telegráfica, telefónica o radiotelegráfica y demás análogos".

> "Transmisión, emisión o recepción de informaciones de cualquier naturaleza por sistemas electromagnéticos".

> "Science and practice of transmitting information by electromagnetic means. A wide variety of information can be transferred through a telecommunications system, including voice and music, still-frame... ('ciencia y práctica de la transmisión de información por medios electromagnéticos. Una gran variedad de informaciones se pueden transferir por medio de los sistemas de telecomunicación, entre ellas la voz, la música, etc.)".

Tal como podemos comprobar, todos coinciden, con más o menos matices, en definir la telecomunicación en su manera más básica como transmisión de información o comunicación a distancia y matizan el medio. Hay quienes dicen *medios electromagnéticos*, otros especifican *telefonía*, *telegrafía*, etc.

Los primeros sistemas de comunicación a distancia los encontramos con los mensajeros, las famosas señales de humo de los indios, etc. ¿Podemos considerar estos sistemas como los primeros sistemas de telecomunicación? Por lo que hemos visto en las definiciones, no. La *Encyclopedia Britannica* nos deja claro que se deben utilizar métodos electromagnéticos para transmitir la información para poder considerar que estamos ante un sistema de telecomunicación, y la RAE directamente afirma que deben ser sistemas telegráficos, telefónicos, de radio o similares. Con esta restricción del abanico de posibilidades debemos avanzar mucho en la línea del tiempo para poder encontrar los primeros sistemas de telecomunicaciones. El telégrafo, las primeras emisiones de radio, los primeros teléfonos, etc. utilizan ya los principios del electromagnetismo desarrollados por Maxwell para transmitir la información.

James Clerk Maxwell

James Clerk Maxwell (1831-1879) consiguió expresar las leyes descubiertas por Coulomb, Faraday y Ampère en un conjunto de ecuaciones (ecuaciones de Maxwell) que relacionan matemáticamente las distribuciones de carga y de corriente con las fuerzas eléctricas y magnéticas que generan. Cualquier problema electromagnético se puede resolver utilizando las ecuaciones de Maxwell.

Satélite geoestacionario

El satélite geoestacionario es un satélite con órbita geoestacionaria. La órbita geoestacionaria está a 36.000 km de la Tierra y tiene la virtud de que el satélite se mueve a la misma velocidad angular de rotación de la Tierra, de manera que desde la superficie de ésta parece que no se mueva de sitio.

En la actualidad no nos sería nada difícil encontrar sistemas de telecomunicación a nuestro alrededor, probablemente el lector tiene en estos momentos bien cerca de él un teléfono móvil, un teléfono fijo, un ordenador conectado en red con otros ordenadores, con acceso a Internet, quizá una línea ADSL, y seguro que no muy lejos también, un televisor, una radio e, incluso, en el techo del edificio quizá haya una antena parabólica capaz de recoger eficazmente señales de algún satélite geoestacionario. Quizá incluso tiene un coche equipado con navegador, que conecta con el sistema GPS[1] para dar la posición o seguir una ruta concreta. Es evidente que los sistemas de telecomunicación se han convertido en compañeros cotidianos de todos nosotros y que incluso a muchos les resultaría difícil mantener su actividad profesional e incluso de ocio sin éstos.

(1)GPS: *global position system*; sistema formado por 24 satélites de órbita baja que nos dan con precisión la posición del receptor sobre la superficie terrestre.

Independientemente de la simplicidad o de la complejidad que tengan los sistemas de telecomunicación, para que la comunicación les sea posible a todos, podemos identificar unas partes comunes e independientes de la tecnología del sistema. Es decir, en todo sistema de telecomunicación podemos distinguir las partes esenciales de toda comunicación: emisor, medio, mensaje, código, receptor.

1.1.1. Elementos de la comunicación

Toda comunicación, sea de la naturaleza que sea, para que tenga éxito, debe incluir un emisor, un medio de transmisión, un mensaje para transmitir, un código conocido por los interlocutores y un receptor. El mensaje tiene que responder a un código conocido tanto por el transmisor como por el receptor.

- **Emisor**: es el que proporciona la información.

- **Medio**: es el vehículo por el cual viaja la información.

- **Mensaje**: es el total de la información que proporciona el emisor.

- **Código**: es el conjunto de reglas y normas que deben conocer tanto el emisor, como el receptor con el fin de poder tratar el mensaje.

- **Receptor**: es el que recibe la información.

1.1.2. Elementos en un sistema de telecomunicación

Todo sistema de telecomunicación, sea cual sea la tecnología utilizada, tiene como objetivo comunicar al emisor y al receptor; por lo tanto, debe contar con los mismos elementos que una comunicación (un emisor, un medio de transmisión, un mensaje para transmitir, un código conocido por los interlocutores y un receptor).

En el caso de un sistema de telecomunicación se introduce la particularidad de que la información se emitirá, viajará y se recibirá mediante sistemas electromagnéticos.

Es decir, ahora se pueden definir las partes añadiendo este matiz:

- **Emisor**: es el que proporciona la información modelada para ser transmitida en forma de impulsos electromagnéticos.

- **Medio**: es el vehículo por el cual viaja la información electromagnética.

- **Mensaje**: es el total de la información que proporciona el emisor. Dentro del sistema de telecomunicación el mensaje irá disfrazado de lo que llamaremos *señal*.

- **Código**: es el conjunto de reglas y normas que deben conocer tanto el emisor como el receptor con el fin de poder tratar el mensaje.

- **Receptor**: es el que recibe la información en forma de impulsos electromagnéticos.

La obsesión de todo sistema de telecomunicación es conservar, mimar, y tener el máximo de cuidado de la señal que viaja por el medio para que no se degrade y, de este modo, poder hacer llegar al receptor toda la información. La calidad de todo sistema de telecomunicación depende de la capacidad que tenga el receptor de reproducir la señal que le llega fielmente, de la forma en la que se ha emitido.

1.2. La señal en un sistema de telecomunicación

Como ya hemos apuntado en el apartado anterior, todos los elementos del sistema tienen como objetivo común mantener la información veraz. La información viaja, es el pasajero de la señal, el objetivo común es hacer llegar al receptor una señal idéntica a la que se ha emitido al emisor. La señal sólo llegará idéntica en casos ideales, en los casos reales siempre habrá alguna variación respecto a la señal emitida. El sistema puede permitir que la señal cambie formalmente, pero es necesario que el viajero de la señal, el mensaje, sea igual o al menos mantenga su significado idéntico a como se ha emitido.

En función de la tecnología del sistema de telecomunicación, la señal se presentará en forma de ondas de radio, luz, señales eléctricas, etc. El factor común de todos es que físicamente tienen una base electromagnética.

Para el curso, no nos interesan las bases electromagnéticas de la señal ni tampoco queremos realizar un modelo matemático que explique el comportamiento y las propiedades, en cambio sí que conviene saber qué tipos de señal se pueden tener.

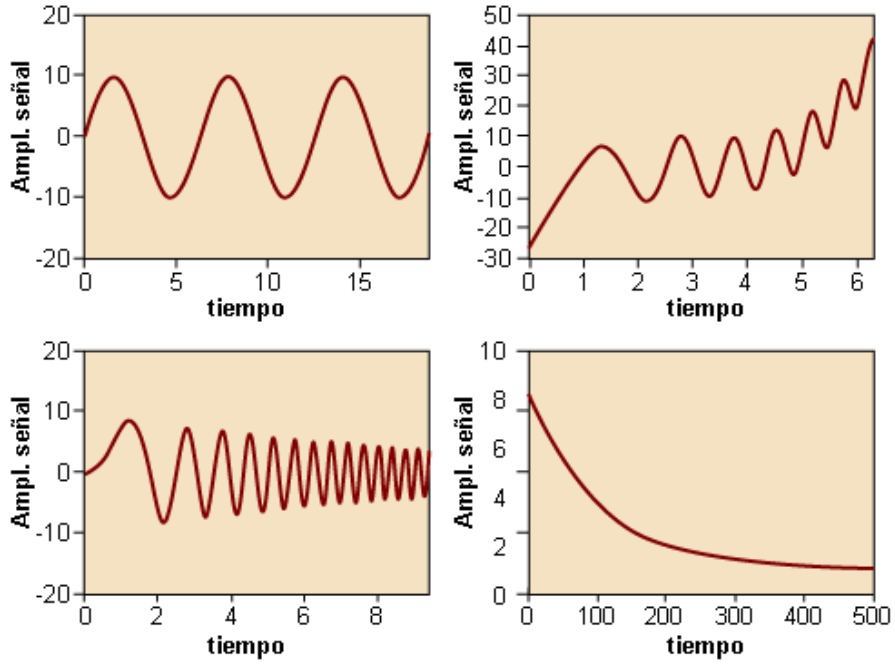
1.2.1. Tipos de señal

Se puede tipificar la señal basándose en muchos criterios, pero a nosotros sólo nos interesa tener clara la diferencia entre una señal analógica y una señal digital. Esta diferencia condiciona todo el sistema, ya que los sistemas analógicos son absolutamente incompatibles con los sistemas digitales y viceversa.

Señal analógica

Por definición, una señal analógica es toda señal continua en el tiempo.

Para poder ver más claro el concepto que la pura definición, nos será muy útil representar señales gráficamente en unos ejes cartesianos donde el eje de ordenadas sea la intensidad de la señal y el eje de abscisas sea el tiempo.



Ejemplos de señales analógicas.

Como podemos ver en los gráficos anteriores, todas las señales son continuas a lo largo del tiempo, no hay ninguna interrupción de la señal.

Nota

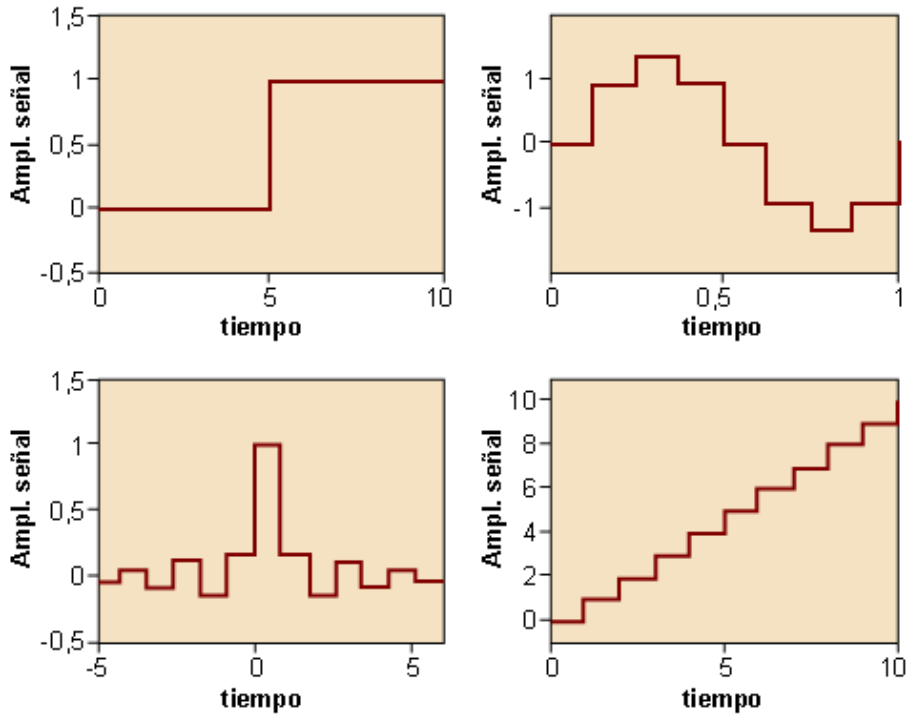
En la naturaleza todas las señales son analógicas.

Señal digital

En contraposición a la señal analógica, la señal digital se define como aquella señal que es discreta en el tiempo.

Una señal digital no puede tener continuidad en el tiempo, de manera que una señal nunca podrá ser analógica y digital a la vez.

Igual que en el caso de la señal analógica, su representación gráfica nos ayudará a entender el concepto.



Ejemplos de señales digitales.

Como se puede ver en los gráficos anteriores, las señales de ejemplo no son continuas en el tiempo, van dando saltos. Este cambio repentino de un valor a otro es el que no puede hacer nunca una señal analógica.

En la naturaleza no hay señales digitales, todo son señales analógicas. Aunque mediante generadores de señales podemos conseguir peldaños muy similares a los de las gráficas, el salto nunca tendrá una pendiente vertical como en las gráficas.

Función no derivable

Matemáticamente se dice que la función no es derivable en los puntos donde se producen las discontinuidades, los saltos.

1.3. El ruido en un sistema de telecomunicación

Dentro de un sistema de telecomunicación entendemos como ruido toda aquella señal indeseada que perturba la señal que contiene la información, independientemente de su naturaleza.

1.3.1. Fuentes de ruido

El origen de las fuentes de ruidos puede ser muy variado y no nos preocupa porque a menudo no podemos hacer nada sobre la fuente, lo que nos preocupa son sus efectos sobre la señal de información y es necesario que el sistema se proteja contra estos efectos.

Hay ruido de origen **natural**, de origen **humano** y de origen **cósmico**. Para el sistema, el origen no es importante porque los efectos siempre son los mismos, degradar la señal. Son fuentes de ruido:

- **La temperatura**: un cable, únicamente por tener una temperatura determinada, hace que su estructura atómica radie pequeñas cantidades de energía que se pueden superponer a la señal de información. Cuanto más alta sea la temperatura, más radiación existe y más nocivos resultan los efectos.

- **Cualquier otra emisión deradiofrecuencia**: se pueden superponer a la emisión de la señal de información. Que los efectos sean o no nocivos dependerá de si las dos señales comparten alguna frecuencia (si los espectros se encabalgan).

- **El sol**: una antena orientada al sol provoca que aparte de la señal de información reciba una gran radiación de otras señales que evitan que sea capaz de discriminar correctamente la señal de información. Igual que le sucede al ojo humano, que se deslumbra al mirar el sol y no ve bien (desde el punto de vista que nos ocupa, se puede decir que el ojo humano es una antena excelente que tiene por ancho de banda las frecuencias que hay en el espectro visible de la luz).

- **Campos magnéticos**, tanto de origen natural como los causados por la actividad humana, tales como las líneas de alta tensión, motores eléctricos, etc., también pueden afectar a la señal de información.

La calidad del sistema, si tomamos como referencia el ruido, nos la proporciona la relación señal-ruido. Se estudiará la relación señal-ruido después del apartado de procesamiento de la señal, ya que así se tendrá un criterio mejor para determinar si una señal digital es más o menos resistente al ruido.

1.4. Procesamiento de la señal

Vemos que la señal es una entidad débil y atacada constantemente por el ruido, por lo tanto, es necesario que dotemos al sistema de formas para recuperar la salud, ya que de ello depende que la información llegue al receptor fiel como se la ha dado el emisor.

En un sistema se pueden instalar muchos elementos que ayuden a rehacer la señal si se ha degradado, pero esto siempre queda a criterio de quien diseñe y gestione el sistema. El primer paso, y el más importante, es proporcionar al sistema de telecomunicación una señal fuerte, capaz de luchar por sí misma contra todas las circunstancias que se encuentre en el camino. Por este motivo no sirve cualquier señal.

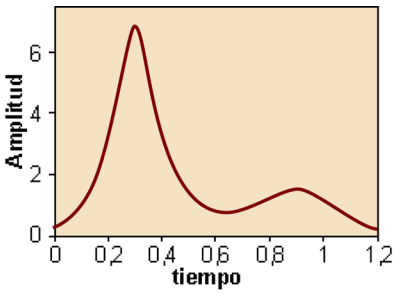
Hay señales que por naturaleza están mejor dotadas para ser usadas en un sistema de telecomunicaciones, son más resistentes al ruido. Es obligatorio que la señal que contiene la información tome una de estas formas sin que la información que viaja dentro de la señal se altere.

Será bueno manipular, transformar y adaptar el mensaje con la información hasta que tome la forma óptima para ser transmitida. Todas estas manipulaciones de la señal se conocen como *procesamiento de la señal*.

1.4.1. Digitalización de la señal

Como se puede comprobar a lo largo de esta unidad, por su propia naturaleza, una señal digital es más resistente al ruido que una señal analógica; por lo tanto, se recomienda utilizar sistemas de transmisión digital con el fin de contar con las máximas garantías de recibir correctamente la señal. Por otra parte, también debemos recordar que en la naturaleza todas las señales son analógicas y, en consecuencia, se debe hacer un primer procesamiento de la señal para convertirla en digital.

Para ilustrar el paso de analógica a digital (digitalización) partiremos de la señal siguiente, que a partir de ahora denominaremos *DEMO*:



Señal analógica. En el resto del capítulo nos referiremos a esta señal como DEMO (demostración).

Si la diferencia entre una señal digital y una analógica es su continuidad en el tiempo, para pasar de analógica a digital lo primero que se debe hacer es romper su continuidad en el tiempo. Este proceso se denomina *muestreo*.

Muestreo

Se trata de tomar muestras de la señal analógica. Estas muestras se deben tomar equidistantes unas de otras, es decir, hemos de tomar las muestras de la amplitud de la señal en intervalos de tiempos iguales. El tiempo entre muestras se denomina *período de muestreo* (T_m) o frecuencia de muestreo ($f_m = 1 / T_m$).

No se puede elegir un período de muestreo al azar, si lo hiciéramos correríamos el peligro de tomar pocas muestras (submuestreo) y no poder reproducir fielmente la señal en el receptor por falta de información. También nos podría ocurrir que tomáramos más muestras de la cuenta (sobremuestreo). En este ca-

so no habría problemas de fidelidad en la señal pero estaríamos transmitiendo más información de la necesaria y el sistema perdería eficiencia (ocuparíamos un ancho de banda innecesario).

¿Cómo se puede elegir un período de muestreo correcto? Sólo debemos respetar la ley de Nyquist. Esta ley nos advierte que:

"La frecuencia de muestreo debe ser, como mínimo, dos veces el ancho de banda[2] de la señal, $F_m >= 2 W$".

Ley de Nyquist

La desarrolló el físico e ingeniero eléctrico y de comunicaciones Harry Nyquist. Nació en Suecia el 7 de febrero de 1889 y murió en Texas el 4 de abril de 1976.

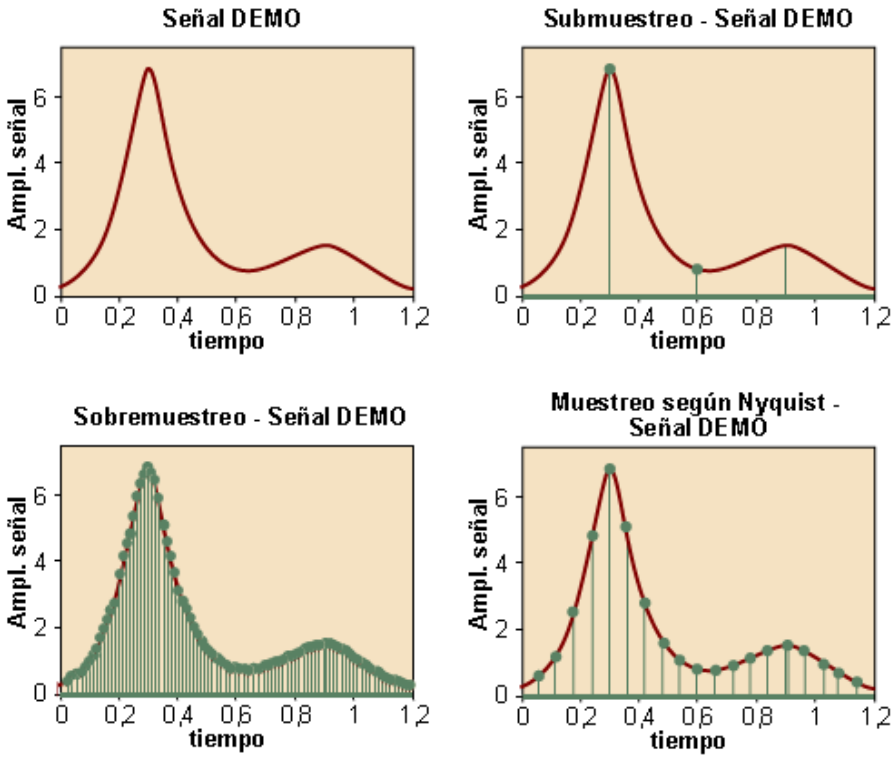
Para saber más

Para disponer de más información sobre el ancho de banda, podéis consultar el apartado "Ancho de banda" de la unidad "Dimensionado de redes".

(2)Rango de frecuencias que ocupa la señal.

Esto nos dice que debemos tomar muestras con una frecuencia mínima dos veces superior a la frecuencia máxima de la señal.

Para asentar los conceptos anteriores nos pueden ayudar los gráficos siguientes, donde se ve el proceso de muestreo de una señal analógica presentada anteriormente (señal DEMO).



Muestreo de la señal DEMO.

En los gráficos anteriores se pueden ver tres casos de muestreo de la función DEMO.

Valor de ancho de banda supuesto

No es real para poder dibujar las muestras en los gráficos con claridad.

Para saber cuál es el muestreo ideal, nos es necesario saber el ancho de banda o frecuencia máxima de la señal. Averiguar el ancho de banda de la señal requiere un análisis espectral con fuerte contenido matemático que está fuera de los objetivos del curso. De todas modos, supondremos que el ancho de banda de la señal es de 16,5 Hz.

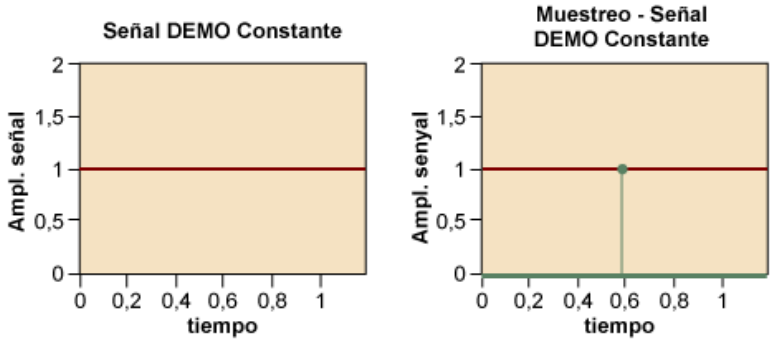
En la gráfica superior derecha tenemos un claro submuestreo. Independientemente de saber o no el ancho de banda, ya se puede intuir que con sólo estas cinco muestras nos será imposible reproducir fielmente la señal DEMO. El muestreo recoge cinco muestras, es decir, tomamos muestras cada 1,2/4 = 0,3 s, T_m = 0,3 s, f_m = 1/0,3 = 3,33 Hz ($f_m < f_{nyquist}$).

En la gráfica inferior izquierda tenemos un claro sobremuestreo. Independientemente de saber o no el ancho de banda, podemos intuir que estamos recogiendo un número exagerado de muestras, más de los estrictamente imprescindibles. El muestreo recoge 201 muestras, es decir, tomamos muestras cada 1,2/200 = 0,006 s, T_m = 0,005 s, f_m = 1/0,006 = 166,67 Hz ($f_m >> f_{nyquist}$).

En la gráfica inferior derecha tenemos el muestreo correcto según Nyquist. Sabiendo que el ancho de banda de la señal es de 16,5 Hz, debemos tomar muestras en un período no superior a 1/16,5 = 0,061 s. El muestreo se ha realizado con un período de 0,06 s, de modo que en el intervalo de tiempo de la gráfica se han recogido 21 muestras. Se toman muestras cada 1,2/20 = 0,06 s, T_m = 0,06 s, f_m = 1/0,06 = 16,67 Hz ($f_m > f_{nyquist}$).

Señal y frecuencia

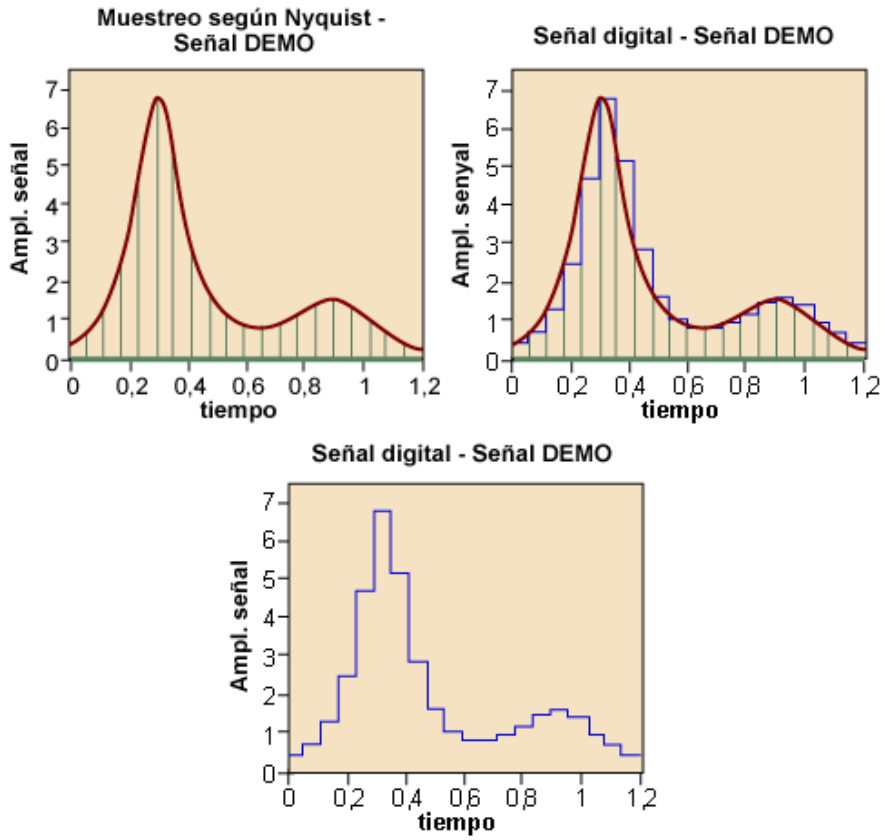
Como se puede intuir, cuanta menos variación tenga una señal, menos frecuencia máxima tiene y menos muestras es necesario tomar por unidad de tiempo. Llevando esto al límite, tenemos que una señal constante tiene frecuencia cero, de manera que tomando una sola muestra ya cumplimos el criterio de Nyquist.



Con la única muestra tomada en la gráfica de la derecha podemos reproducir exactamente la señal de la gráfica de la izquierda, ya que al contar con la amplitud constante sólo debemos saber, precisamente, el valor de la amplitud.

Cuantificación

Una vez realizado el muestreo, ya podemos crear la señal digital, aunque es una señal digital demasiado compleja para ser manipulada. La complejidad recae en el hecho de que si bien las muestras nos dan una señal discreta en el tiempo, el valor de cada muestra puede tomar un número infinito de valores. Nos gustaría acotar este número a unos pocos valores.



Señal digital obtenida de las muestras de la señal DEMO.

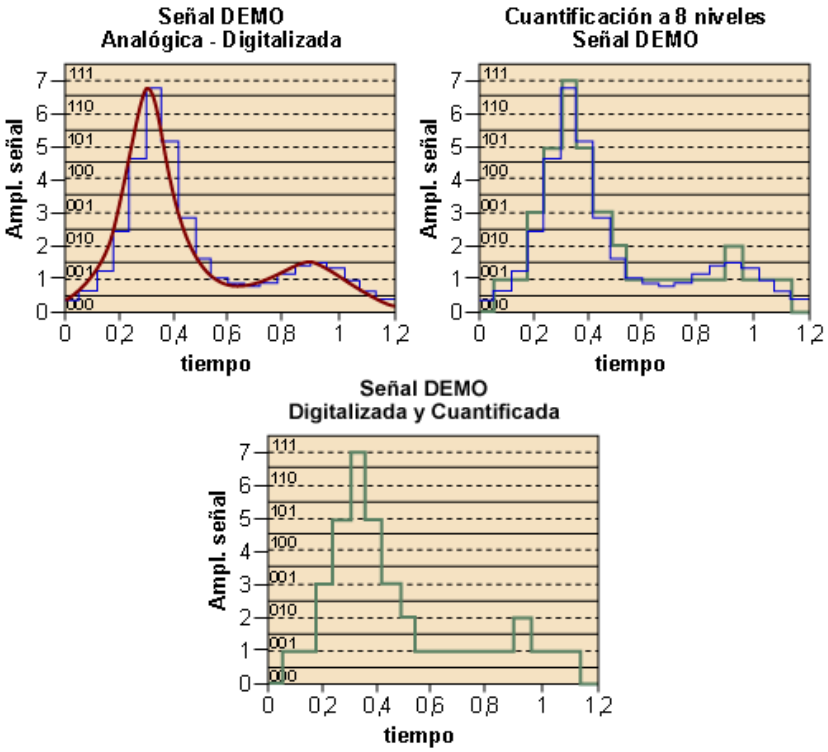
En la gráfica superior derecha vemos cómo a partir de las muestras creamos la señal digital. Veamos más claramente la señal digital en la gráfica inferior.

El objetivo es limitar la amplitud de cada peldaño de la señal digital, de manera que tome un número finito y conocido de valores. Para hacerlo habrá que decidir cuántos y qué valores permitimos que tome la señal. Una vez que sabemos estos valores, nos definimos unos umbrales de decisión con el fin de restringir cada muestra al valor permitido que más se le acerque.

Para nuestro ejemplo, permitiremos ocho valores: 0, 1, 2, 3, 4, 5, 6 y 7, de manera que los peldaños que estén por debajo de la amplitud 0,5, los consideraremos 0, a los peldaños con una amplitud entre 0,6 y 1,5 se les asigna el

valor 1, entre 1,6 y 2,5 se asigna valor 2, y así sucesivamente hasta transformar nuestra señal digital en otra señal digital pero ahora con su amplitud restringida a un número acotado de valores.

En los gráficos siguientes se ve el proceso de la cuantificación con detalle.



Cuantificación de las muestras obtenidas de la señal DEMO.

En la gráfica superior izquierda podemos ver la señal analógica y la señal digital fruto del muestreo. En la gráfica se han marcado con líneas discontinuas horizontales los únicos valores que queremos que tome la señal digital. Con líneas continuas se han marcado los umbrales de decisión para determinar si una muestra (peldaño de la señal digital) pertenece a un valor permitido u otro.

En la gráfica superior derecha está la clave para entender la cuantificación. Vemos la señal digital anterior (color azul) y la señal digital nueva (color verde), la que resulta de transformar la altura de cada peldaño a su correspondiente valor permitido.

Finalmente, en el gráfico inferior tenemos la señal digital cuantificada que ha resultado.

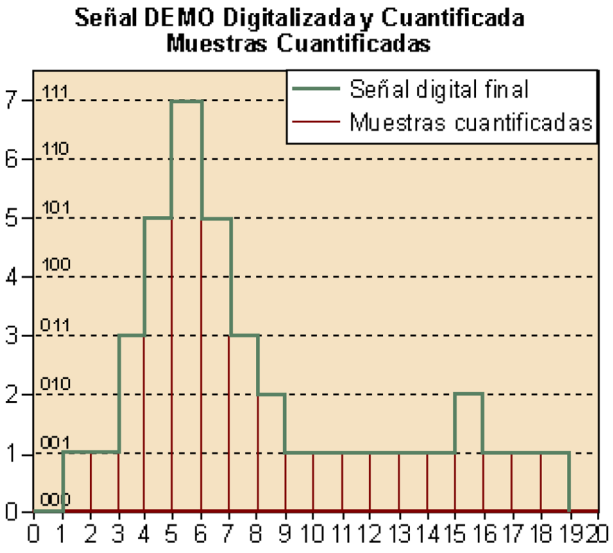
Desde el principio, lo que buscamos es transmitir la información de la señal DEMO en forma de unos y ceros. La respuesta final a esta intención la encontramos en los niveles de cuantificación. Tenemos 8 niveles de cuantificación

y en binario para representar el ocho necesitamos tres bits ($2^n = 8$, n = 3). Por ello, a la izquierda de cada gráfica, sobre cada una de las líneas discontinuas que marcan los niveles permitidos está escrito el valor del nivel en binario.

Tabla de conversación decimal-binario

Sist. decimal	Sist. binario
0	000
1	001
2	010
3	011
4	100
5	101
6	110
7	111

Por lo tanto, podemos escribir la señal como en el tren de bits siguiente:



Senyal DEMO digital.

Es decir, transmitir la señal DEMO equivale a transmitir el tren de bits siguiente que nos marca la segunda fila de la tabla anterior.

Tabla. Relación decimal-binaria de los niveles cuantificados

Muestra núm.	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
Valor en bits	000	001	001	011	101	111	101	011	010	001	001	001	001	001	001	010	001	001	001	000	000

Cuando cuantificamos, inevitablemente perdemos información (ya que cambiamos la altura de las muestras obtenidas de la señal original). ¡Esta pérdida será irrecuperable! El receptor nunca podrá saber cuál era la señal analógica

generada en el emisor con un 100% de precisión. Evidentemente, cuantos más niveles de cuantificación tengamos, menos severa será la pérdida de fidelidad. El precio que se debe pagar siempre es el ancho de banda, se han de transmitir más bits por segundo, más información.

La pérdida de esta fidelidad no debe alarmar, ya que, si bien el mensaje formalmente no es fiel al 100%, su significado sí que puede ser perfectamente inteligible. La telefonía es un ejemplo en este sentido; todos hemos comprobado que la voz de las personas cambia si la escuchamos vía telefónica, y eso no es ningún problema para que los interlocutores puedan entender sus mensajes.

1.4.2. Codificación

Una vez realizado el muestreo y la cuantificación, obtenemos una colección de bits (unos y ceros). Con el fin de hacer más eficiente la transmisión y enviar más información al mismo tiempo, se puede romper la colección y hacer grupos de 2, 3, 4 o n bits. De este modo se puede codificar cada grupo y enviar un único símbolo que represente al grupo.

Si hacemos grupos de 2 bits, tendremos 4 símbolos posibles: 00, 01, 10, 11. Si el grupo es de 3 bits, contamos con 8 símbolos posibles: 000, 001, 010, 011, 100, 101, 110, 111. De manera general, para un grupo con n bits tenemos 2^n símbolos.

Obviamente, el receptor ha de ser capaz de distinguir entre 2^n símbolos con el fin de realizar el paso inverso correctamente.

Volviendo al ejemplo de la señal DEMO, supongamos que nuestro sistema digital es capaz de discriminar no sólo entre dos valores (unos y ceros), sino que es capaz de diferenciar 8 valores (A, B, C, D, E, F, G, H).

Si hacemos la asociación de los ocho valores con los ocho niveles con los que se ha cuantificado:

Tabla de conversión de bits a baudios (símbolos)

Valores binarios de la cuantificación	Símbolos para transmitir
000	A
001	B
010	C
011	D
100	E
101	F
110	G

Valores binarios de la cuantificación	Símbolos para transmitir
111	H

En este caso, el tren de símbolos que se debe transmitir sería:

Relación decimal-binario-baudis (símbolos) de los niveles cuantificados

Muestra núm.	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
Valor en bits	000	001	001	011	101	111	101	011	010	001	001	001	001	001	001	010	001	001	001	000	000
Símbolo transmitido	A	B	B	D	F	H	F	D	G	B	B	B	B	B	B	C	B	B	B	A	A

ABBDFHFDGBBBBBBCBBBAA

Como se puede ver, al codificar se envían menos datos pero el mismo mensaje. Por lo tanto, se gana en eficiencia del sistema. El precio que se paga es que los emisores y receptores son más complejos, más caros y la señal se hace menos resistente al ruido cuantos más niveles sean capaces de discriminar.

En general, cuanto mayor sea el número de símbolos, es más probable que el ruido acumulado en la señal consiga cambiar un símbolo por otro.

Cuando se emiten símbolos, se habla de baudios y, a menudo, la velocidad de transmisión se da con baudios por segundo (equivale a símbolos por segundo). Si cada baudio tiene tres bits, la velocidad en bits (bps) será tres veces la velocidad en baudios. Actualmente, ya casi ha desaparecido el uso del término baudios, ya que la dependencia que tiene con la codificación hace que no dé una idea real de la velocidad de la transmisión. La única unidad de magnitud que nos da la velocidad real de transmisión son los bits por segundo (bps).

1.4.3. Modulación

Modular la señal es uno de los pasos más importantes en la lucha contra el ruido; de la calidad de la modulación depende, en buena medida, la eficacia final del sistema.

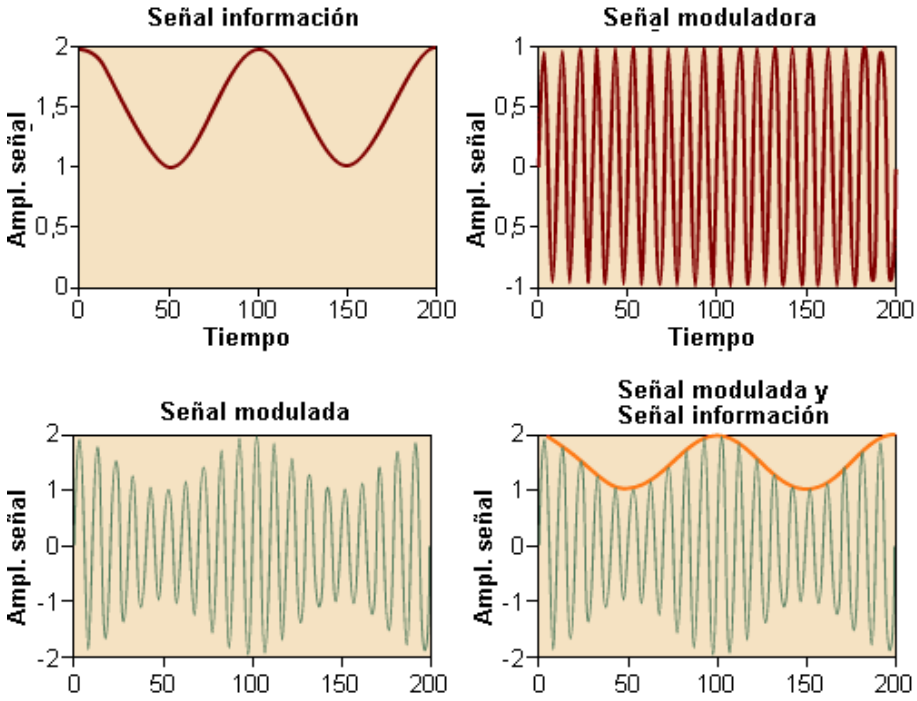
Hay señales que, por naturaleza, son muy resistentes a interferencias y ruidos. Hace falta aprovechar esta característica y usar estas señales para la transmisión. La modulación consiste en poner la señal a mano, la información dentro de una de estas señales capaces de viajar con un grado de degradación mínimo.

Modulación analógica

Empezaremos explicando el concepto de modulación por una señal analógica porque es mucho más intuitivo que el caso digital. Como ya se ha apuntado en la introducción, la filosofía de la modulación es la de poner la señal de información dentro de otra señal, que sabemos que es la óptima para ser transmitida, por su resistencia a las interferencias, por su poca atenuación, etc.

En toda modulación tenemos dos señales: la señal moduladora, a veces también denominada portadora, y la señal para modular o la señal con la información para transmitir. Una vez hecha la modulación, obtenemos la señal modulada, que es una señal formalmente con la apariencia y las propiedades de la señal moduladora pero que, dentro de ella, contiene la señal de información. De manera que, al modular la señal de información, ésta viaja como pasajero de un vehículo denominado señal modulada y que ha aparecido gracias a la señal moduladora.

En los ejemplos de los gráficos siguientes encontramos la modulación de una señal analógica:



Proceso de modulación de amplitud o modulación AM.

En la gráfica superior izquierda tenemos la señal que queremos transmitir, la que contiene la información.

En la gráfica superior derecha aparece una señal portadora. Sus propiedades hacen que sea una señal mejor para transmitir.

En la gráfica inferior izquierda encontramos el resultado de la modulación. Se han multiplicado las dos señales de las figuras superiores y ha resultado una señal modulada, contiene la frecuencia de la señal portadora y dentro viaja la señal que contenía la información que se debía transmitir. La señal de información se encuentra en el envoltorio de la señal modulada, como se puede ver en el gráfico inferior derecho.

Modulaciones

Se ha visto un ejemplo de modulación de amplitud (AM). La información viaja en la amplitud de la señal portadora. Cada sistema tiene sus propias modulaciones; FM (frecuencia modulada), donde la información modula la frecuencia, modulaciones de lado vestigial, que utilizan los sistemas de televisión, etc.

Los sistemas digitales tienen sus propias modulaciones, pero en general son mucho más complejas que la modulación AM vista en los gráficos.

Modulación digital

Recordamos que, por las señales digitales, lo que tenemos que transmitir son símbolos. En el caso más básico, un símbolo equivale a un bit, es decir, un 1 o un 0, la señal tendría la forma de un tren de pulsos cuadrados.

La señal moduladora o portadora será la misma que en el ejemplo analógico, ya que sabemos que es una señal óptima para la transmisión.

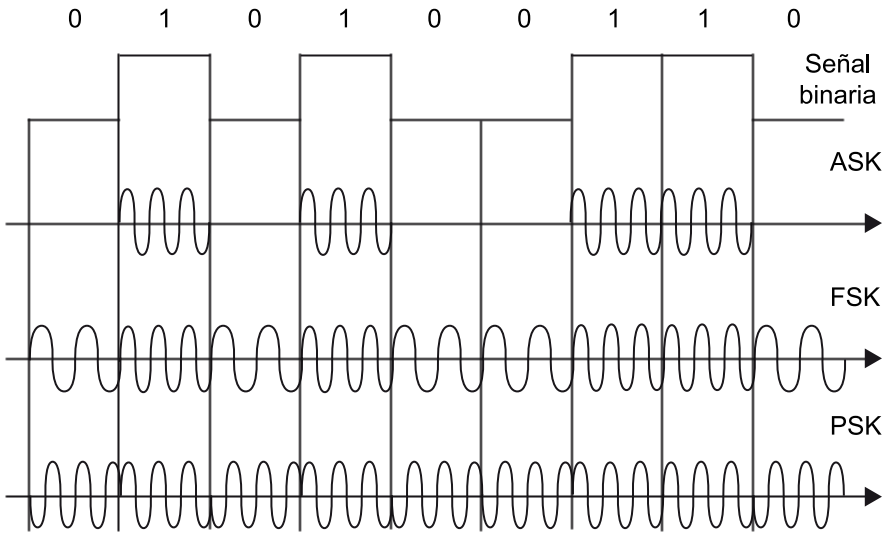
A diferencia del caso analógico, no multiplicaremos las dos señales, sino que usaremos la señal moduladora para indicar si hay que transmitir un 1 o un cero.

Hay, básicamente, tres posibilidades:

- Modular la amplitud - ASK (*amplitude shift keying*): en este caso, los 0 se transmiten como ausencia de señal, y los 1 como presencia de señal.

- Modular la frecuencia - FSK (*frequency shift keying*): en este caso, los 0 y los 1 se simbolizan con la misma señal, pero con una frecuencia ligeramente diferente.

- Modular la fase - PSK (*phase shift keying*): en este caso, los 0 y los 1 se simbolizan con la misma señal, pero en contrafase.



Proceso de modulación digital.

Para el caso de una codificación de la señal con M símbolos, se pueden extrapolar los criterios, lo que daría lugar a las modulaciones M-ASK, M-FSK, M-PSK.

1.4.4. Demodulación

Es el proceso inverso a la modulación. Se trata de rescatar la señal de información del interior de la señal modulada.

Para el caso particular de una modulación AM, el demodulador debe ser un sistema capaz de captar el envoltorio de la señal modulada, que coincide con la señal de información.

Como hemos visto en la gráfica inferior izquierda "Proceso de modulación de amplitud o modulación AM" del apartado anterior, el envoltorio contiene la información útil.

1.4.5. Cifrado y encriptación

Cifrar o encriptar un mensaje nos garantiza la confidencialidad de los datos que se deben transmitir. Suelen ser elementos que siguen algoritmos muy complejos para que el mensaje se pueda descifrar únicamente por el destinatario que posea la clave. Sin embargo, se puede demostrar matemáticamente que el cifrado perfecto no existe; todos se pueden descifrar sin necesidad de conocer la clave *a priori*, a pesar de que a menudo el tiempo que se debe invertir y la potencia de cálculo necesaria para hacerlo son tan grandes que hacen que el proceso sea inviable.

Uno de los métodos más utilizados a la hora de cifrar mensajes consiste en asignar a cada usuario una clave pública y una clave privada con la característica de que la clave particular es la única que puede descifrar los mensajes ci-

frados por su clave pública. Todos conocen la clave pública de cualquier usuario pero sólo el propio usuario conoce su clave privada. El modo de operar es el siguiente:

El usuario A desea enviar un mensaje cifrado al usuario B. Para hacerlo, el usuario A deberá conseguir la clave pública de B. Como es pública, no tendrá ningún problema para conseguirla, cifrará el mensaje utilizando la clave pública de B y lo enviará al usuario B. La única clave que puede descifrar un mensaje cifrado con la clave pública de B es la clave privada de B. Cuando le llegue el mensaje, el usuario B lo podrá descifrar, ya que él es el único que conoce la clave privada de B.

1.4.6. Compresión

En los sistemas de información a menudo los recursos son limitados. Pueden estar limitados por diferentes factores:

- **Espacio**: el espacio físico donde se almacena la información nunca es infinito y dispone de una capacidad limitada.

- **Económicos**: los recursos para almacenar la información no son gratuitos.

- **Técnicos**: los recursos para transmitir la información, básicamente el ancho de banda, es un bien escaso y muy caro. Se debe disponer del hardware que soporte suficiente ancho de banda para transmitir cómodamente la información.

Se pueden aligerar notablemente estas limitaciones comprimiendo la información.

Una compresión nunca es gratuita, a veces se paga con una reducción de la eficiencia y de la calidad del sistema y otras veces se paga sólo con la reducción de la eficiencia del sistema.

Los algoritmos de compresión de información son muchos y muy conocidos, entre los más populares y conocidos se encuentran los archivos ZIP, ARJ, RAR, etc.

Las imágenes, el audio y el vídeo son formatos que ocupan una cantidad enorme de recursos; por este motivo son los archivos más susceptibles de ser comprimidos. Entre las compresiones más conocidas tenemos el algoritmo JPEG para imágenes, MPEG para vídeo y MP3 para audio.

Comprimir con o sin pérdida de calidad

Cada algoritmo de compresión funciona de una manera particular, diferente y a menudo muy astuta.

El ejemplo no quiere constituir un caso real, sólo pretende ilustrar cómo se puede comprimir con o sin pérdida de calidad.

- **Con pérdida de calidad**

Tenemos almacenada en un fichero la imagen siguiente formada por estos 9 píxeles:

En el fichero sin comprimir se encuentra la información contenida de color (V1 = verde1, V2 = verde2, etc.) de cada uno de los píxeles de manera secuencial del primero al último:

Píxel 1	Píxel 2	Píxel 3	Píxel 4	Píxel 5	Píxel 6	Píxel 7	Píxel 8	Píxel 9
V1	V2	V3	B1	B2	B3	R1	R2	R3

El contenido del fichero sería éste:

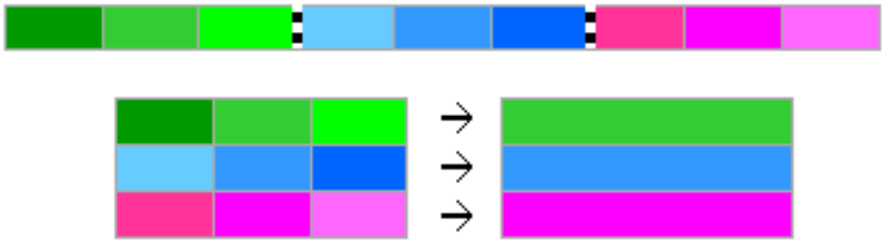
V1V2V3B1B2B3R1R2R3

En el ejemplo, para realizar la compresión se usará la aproximación por patrones o plantillas. El algoritmo haría un estudio de la imagen y extraería un número finito de patrones o imágenes más pequeñas, dividiría la imagen en partes pequeñas y a cada parte le asignaría el patrón que más se le pareciera.

Para nuestro caso suponemos la tabla de patrones:

Tabla de patrones		
Patrón 1		V
Patrón 2		B
Patrón 3		R

El algoritmo dividirá la imagen en partes, por ejemplo de tres píxeles, y asignará a cada parte el patrón que más se le parezca:



Finalmente, el fichero quedaría como:

VRB (compresión del 66%)

Que nos vendría a decir que los tres primeros píxeles se han aproximado al patrón V, los tres siguientes al patrón B, y los tres últimos al patrón R.

Eso se traduce en la imagen:

Evidentemente, en la imagen comprimida hay una reducción en la calidad.

Uno de los formatos más utilizados de compresión de imágenes con pérdida de calidad es el JPEG (*joint photographic expert group*). Cuando comprimimos en JPEG, se puede ajustar el grado de calidad: a más calidad, menos compresión.

- **Sin pérdida de calidad**

Supongamos la imagen siguiente.

La única manera de comprimir sin perder calidad es evitar transmitir la redundancia de la imagen. Afortunadamente, las imágenes suelen ser muy redundantes.

En el fichero sin comprimir transmitiremos el color de cada píxel.

Píxel 1	Píxel 2	Píxel 3	Píxel 4	Píxel 5	Píxel 6	Píxel 7	Píxel 8	Píxel 9
V	V	V	B	B	B	R	R	R

Quedaría un fichero como éste:

VVVBBBRRR

Si evitamos transmitir la redundancia, podemos reducirlo a:

3V3B3R (compresión del 66%)

Es decir, en vez de guardar el color de cada píxel se pueden guardar todos los píxeles consecutivos que son del mismo color seguidos del valor de este color (3V = tres píxeles de color V). Para nuestro caso, los tres primeros de color verde, los tres siguientes de color azul y los tres últimos de color rosa.

La imagen que sale del fichero 3V3B3R es idéntica a la imagen sin comprimir:

Uno de los formatos más utilizados de compresión de imágenes sin pérdida de calidad es el TIF o TIFF (*tagged image file format*).

El factor de compresión

El factor de compresión no es fijo, siempre depende de cada imagen. Si una imagen tiene un píxel de cada color y los colores cambian aleatoriamente sin seguir ningún patrón, nunca se podrá llevar a cabo una compresión sin pérdida de calidad.

1.5. Relación señal-ruido

Hasta ahora se ha visto que la información viaja a través del sistema de telecomunicaciones mediante la señal, que ataca indiscriminadamente el ruido, lo que afecta a la calidad final del sistema.

1.5.1. Relación señal-ruido en un sistema de telecomunicación analógico

La calidad del sistema vendrá dada por el éxito que tenga la señal a la hora de proteger la información frente a los intentos para degradarla que vendrán de parte del ruido. El juez de esta lucha es la relación señal-ruido, y de su valor depende la calidad del sistema.

En otras palabras, la calidad del sistema se mide del siguiente modo: en el receptor se mide la potencia con la que llega la señal de información y también se mide la potencia del ruido que llega. Se hace el cociente entre estas dos potencias y de aquí se extrae el parámetro denominado *relación señal-ruido* (SNR, *signal noise ratio*), que nos da una idea de la calidad del sistema. Este valor se expresa normalmente en decibelios (dB). Cuanto mayor sea el SNR, mejor calidad tendrá el sistema.

1.5.2. Relación señal-ruido en un sistema de telecomunicación digital

Lo que se ha señalado en el apartado anterior como filosofía se aplica a todo sistema de telecomunicación, pero por su propia naturaleza no tiene sentido obtener la relación señal-ruido de un sistema digital.

El equivalente a la relación señal-ruido en un sistema digital es lo que se denomina *probabilidad de error*. La probabilidad de error nos mide la fiabilidad que tiene el receptor de decidir qué símbolo le ha llegado (una vez leído el próximo apartado se entenderá correctamente qué quiere decir "decidir qué símbolo le ha llegado").

1.5.3. Resistencia al ruido

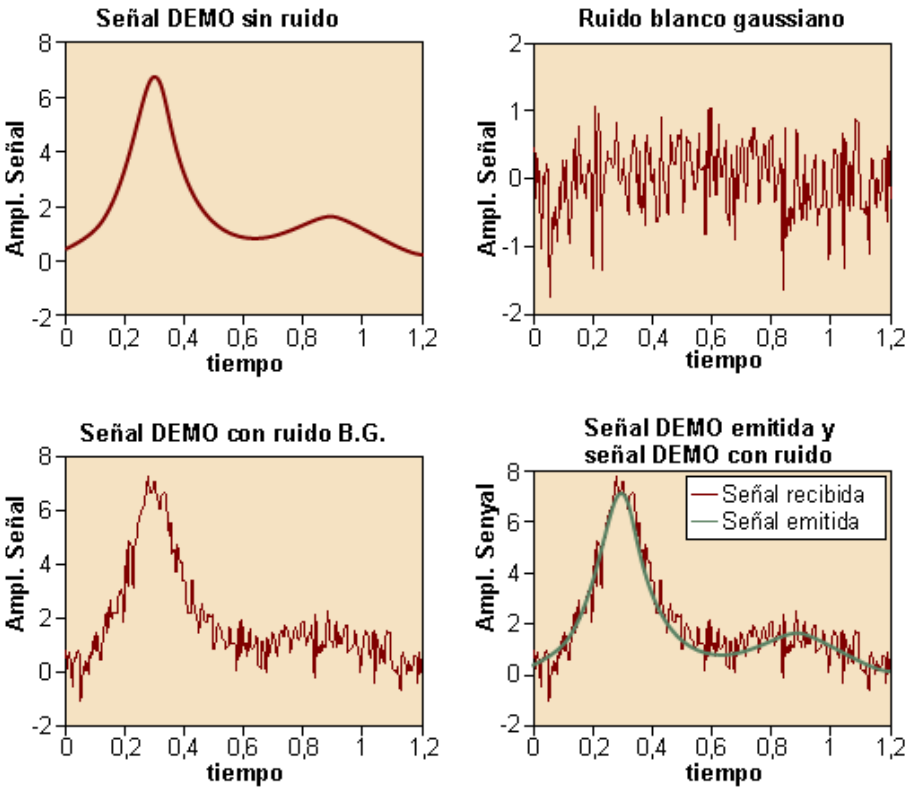
Unas gráficas pueden ayudar a decidir cuál de los dos sistemas, digital o analógico, se comporta mejor ante el ruido.

Sistema analógico

En el conjunto de gráficas inferior se representa la señal DEMO y una función que modela el ruido blanco gausiano. El ruido blanco gausiano es una forma de modelar ruido que se adapta al ruido que capta cualquier antena por el hecho de estar al aire libre.

Televisión sin antena

Si encendemos el TV sin enchufar la antena o bien sin sintonizar ningún canal, veremos una imagen gris muy granulada: el ruido blanco gausiano que capta la entrada de la antena.



Señal DEMO y ruido blanco gausiano.

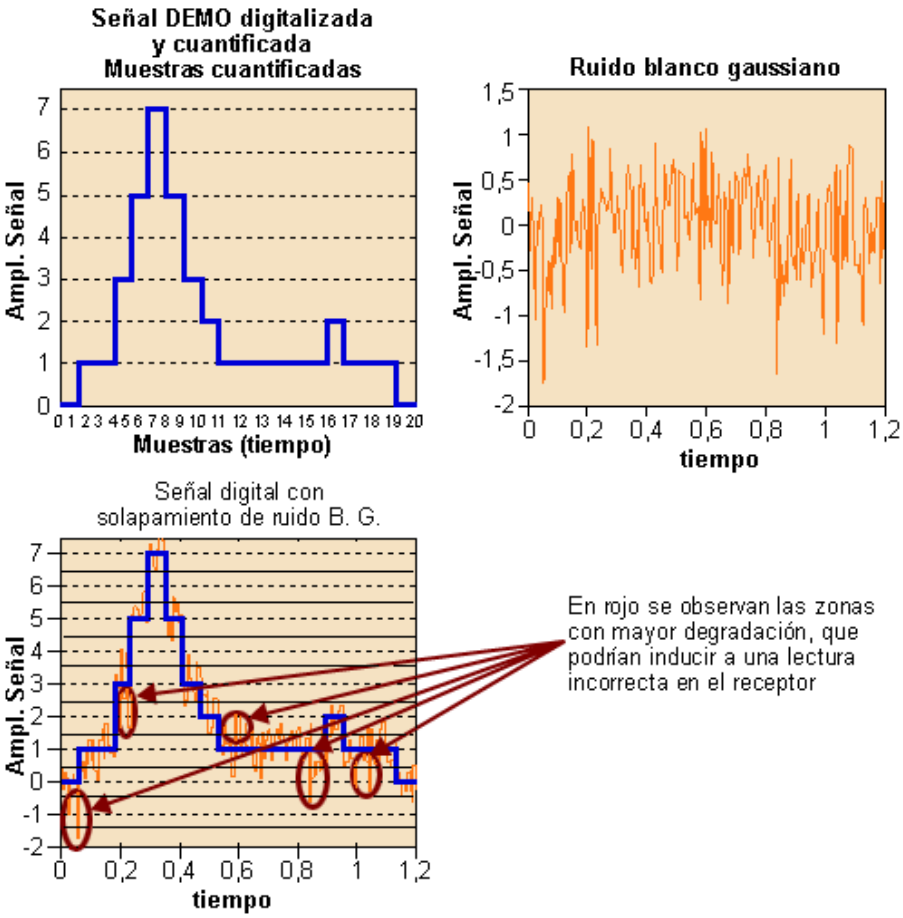
En la gráfica inferior izquierda tenemos la señal que llega al receptor, la señal DEMO y todo el ruido que se le superpone. En la gráfica inferior derecha vemos claramente la diferencia entre la señal en el emisor (señal en azul) y la señal en el receptor (señal en rojo).

Como podemos ver en algunas partes, el ruido ha hecho mucho daño, pero en esencia la señal es muy reconocible y el envoltorio continúa siendo muy similar a la señal DEMO.

Por medios complejos de procesamiento de la señal se puede mejorar y limpiar notablemente el ruido de la señal al receptor. No obstante, no hay manera de limpiar el 100% de la señal recibida.

Sistema digital

En el conjunto de gráficas inferiores se representa la señal DEMO, versión digital. También encontramos la función que modela el ruido blanco gausiano, la misma que se ha presentado para el caso analógico.



Señal DEMO digital y ruido blanco gausiano.

Aparentemente, los efectos son tan desastrosos o más que en el caso analógico. Los efectos son muy perjudiciales porque se ha dibujado un ruido con una potencia muy elevada con el fin de poder ver clara la señal y el ruido.

En realidad, esta señal es más resistente al ruido. El receptor sólo equivocará la lectura cuando el ruido disponga de suficiente amplitud para hacer sobrepasar el umbral, superior o inferior, del nivel que se había transmitido.

Durante todo el estudio se ha supuesto un sistema con 8 niveles de amplitud (0, 1, 2 ... 7). Habíamos visto que los umbrales de decisión estaban situados a (0,5, 1,5, 2,5 ... 7,5) y entre cada nivel tenemos un margen de amplitud igual a 1, 0,5 por arriba y 0,5 por debajo, lo que supone que sólo habrá una mala lectura del receptor si el ruido tiene una amplitud mayor de 0,5 o más pequeña de –0,5. El ruido restante es invisible para nuestro sistema, no se da cuenta de él. Mientras el nivel de ruido no sea más alto de 0,5 o más bajo de –0,5, la señal que nos llega es idéntica a la emitida.

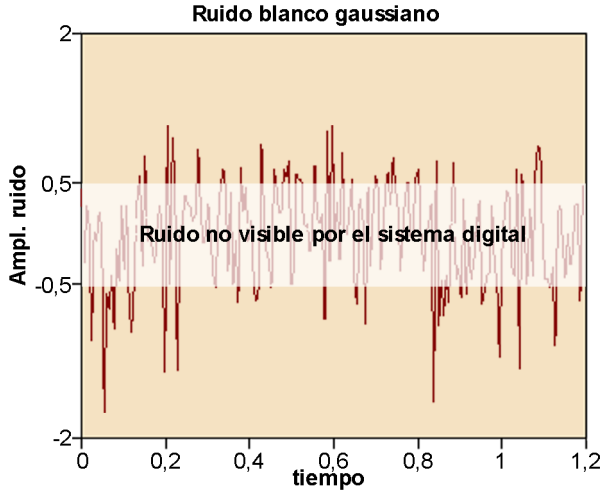
Es fácil ver que, cuanto más niveles, más sensible es al ruido. Cuantos más niveles haya, más se parece a la señal analógica.

Conclusiones; digital frente a analógico

Cuando el ruido ataca un sistema analógico lo degrada, pero éste, aunque degradado, mantiene la información, y, a medida que el ruido se va haciendo más intenso, la información se aleja poco a poco de su forma original.

Cuando el ruido ataca un sistema digital, este sistema digital se mantiene inmune, no ocurre nada hasta que el ruido es tan grande que provoca lecturas erróneas en el receptor.

En el gráfico siguiente se diferencian dos zonas. La zona central del ruido (–0,5 +0,5), donde el sistema digital presenta un rendimiento claramente superior, ya que no le afecta el ruido. Fuera de esta zona, el sistema analógico presenta la información degradada pero para algunos casos suficientemente inteligible; el sistema digital en esta zona hace lecturas falsas de la información.



Ruido blanco gausiano

¿Cuál es mejor? Depende del caso. En el sistema analógico, la información se degrada progresivamente, mientras que en el digital la información del receptor es correcta o es falsa, no hay término medio.

La televisión analógica frente a la televisión digital

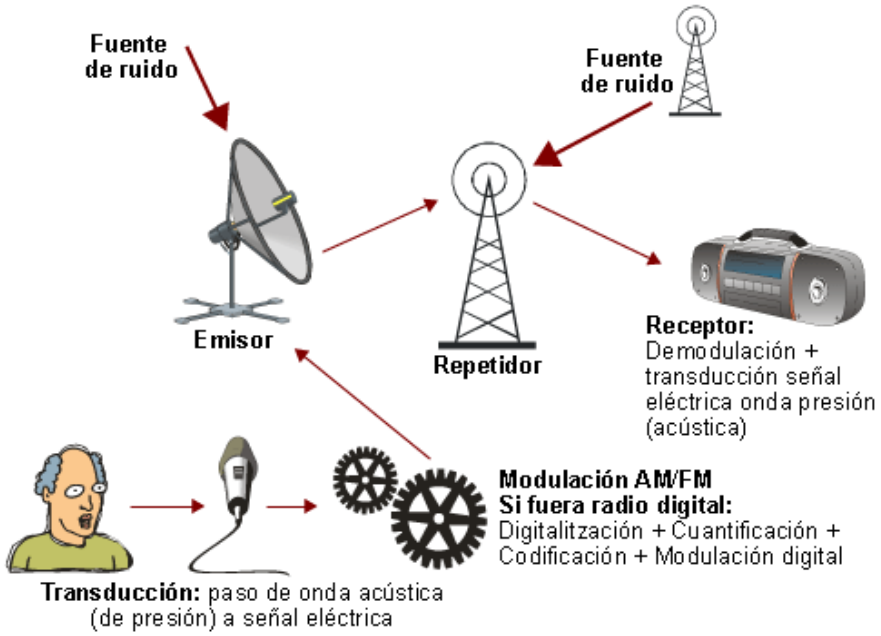
Para ver claramente la comparación utilizamos un sistema conocido: la televisión (TV).

En un receptor de TV analógica, cuando hace mal tiempo o la cobertura de TV es deficiente, los canales se ven con mucha nieve, granulados, con rayas, etc. En general, aunque moleste a la vista se ve la imagen. El ruido degrada y hace que no se vea nítida, pero el espectador puede entender lo que está viendo.

En los sistemas de TV digitales, ya sea vía satélite o vía TDT (televisión digital terrestre), la señal llega al descodificador en forma digital. Cuando por algún motivo, como la meteorología, la señal ha acumulado bastante ruido, vemos que en la pantalla del televisor aparecen cuadritos negros mientras en el resto de la pantalla se ve la imagen perfectamente nítida. Los cuadritos negros corresponden a bits perdidos, zonas donde el ruido ha eliminado la señal y el receptor no puede decidir qué le ha llegado. Ésta es la ventaja y a la vez el inconveniente del sistema digital, se ve perfecto... hasta que no se ve.

1.6. Transmisión de una señal de radio AM/FM

En el gráfico siguiente se pueden ver a alto nivel todos los puntos desarrollados en la unidad para un sistema real de telecomunicación.



Sistema de telecomunicaciones. Emisión de radio AM/FM o digital.

En el gráfico anterior se puede ver cómo la onda acústica emitida por el locutor de radio es recogida por el micro y se pasa a una señal eléctrica (transducción). La señal eléctrica se procesa para convertirla en una señal de radio AM/FM o de radio digital. Se emite en el aire y a partir de aquí la señal viajará por el espacio mientras encuentre repetidores. En este trayecto se irá "infectando" de ruido. Si algún receptor sintoniza la frecuencia en la que se ha modelado la señal, este receptor captará la señal con el ruido que contiene además de otras señales que pueda captar la antena del receptor, que pasarán a ser ruido si el receptor no los filtra suficientemente bien. Finalmente, el receptor procesará la señal captada para extraer la información. Con la información extraída se

efectuará la transducción inversa, la onda eléctrica excitará las membranas del altavoz, que provocará ondas de presión (acústicas) que reproducirán el sonido emitido por el locutor.

2. Red de telecomunicación o informática

Con la proliferación de la informática en las empresas pronto surge la necesidad de conectar los diferentes ordenadores de una sala, de un departamento, de todo el edificio, e incluso de todas sus sedes que la empresa pueda tener tanto dentro como fuera del territorio nacional.

Esta necesidad da lugar a las redes informáticas que hoy día son un elemento imprescindible para cualquier negocio. Las redes informáticas no sólo unen ordenadores, sino que son un elemento esencial para la compartición de recursos, gracias a la red se pueden compartir impresoras, bases de datos, enviar faxes y correos electrónicos desde la estación de trabajo, conectar fotocopiadoras, ordenadores portátiles o trabajar desde algún lugar remoto como si estuviéramos en la oficina. La red estrella desde el punto de vista de la información interna de una organización es la red LAN.

La unidad empieza por definir los diferentes elementos de una red y se da un vistazo al modelo OSI, modelo de referencia para cualquier profesional del ámbito y que hay que conocer al menos descriptivamente para convertirnos en interlocutores competentes a la hora de tratar con proveedores de redes informáticas.

Se presentan las diferentes topologías y los métodos de acceso a la red y, finalmente, se presentan las redes WIFI o sin hilos, que desde hace tiempo han invadido tanto los entornos corporativos como los domésticos.

2.1. Red de telecomunicación o informática. Concepto

Como red de telecomunicación se entiende un concepto muy amplio que agrupa una gran cantidad de redes, si bien todas tienen como factor común el objetivo de distribuir una determinada señal a diferentes redes que pueden tener características muy distintas.

Son redes de telecomunicación: la red de distribución de la señal de televisión, la red de distribución de la señal telefónica, la red de distribución de la señal GPS, la red de distribución de la señal de telefonía móvil, la red de señales de radio FM, la red de señales de radio de bomberos, de policía, de entidades privadas como compañías de transporte, taxis, etc. Es obvio que todas ellas tienen sus características propias y particulares.

A nosotros nos interesa un determinado tipo de redes, muy extendido actualmente, las redes formadas por ordenadores o redes informáticas. En esta unidad estudiaremos algunas de sus características principales.

2.1.1. Elementos de una red informática

La red informática es el soporte físico de un determinado tipo de sistema de telecomunicación y, como tal, incluye los elementos ya vistos de una comunicación, pero también incluye otros que la ayudan para que la red funcione correctamente, es decir, que sea capaz de enviar mensajes de un ordenador a otro o de un ordenador hacia un grupo de ordenadores.

- **Transmisor**. Se denomina *transmisor* el ordenador que emite el mensaje.

- **Medio de transmisión**. Típicamente, el medio de transmisión de una red es el cable, pero también se puede construir una red sin hilos, muy utilizadas en entornos en los que las distancias son cortas, como oficinas, comercios o casas particulares. Al final de la unidad se comentan más detalladamente las redes sin hilos. Los cables más utilizados son:
  - par trenzado,
  - coaxial,
  - fibra óptica.

- **Repetidores**. Estos elementos reciben una señal, generalmente muy débil, y la reenvían con más potencia, de manera que la señal llega al siguiente repetidor o al destinatario final. Tiene el inconveniente de que si la señal llega degradada con ruido o interferencias, también los amplifica y reenvía.

- **Regeneradores**. Actúan como un repetidor, pero son elementos más sofisticados; no se limitan a repetir la señal que les llega, sino que la interpretan, la regeneran y la emiten, de manera que la señal de salida tiene toda la potencia de un repetidor pero, además, es una señal sin degradar, limpia de interferencias y de ruido.

- **Direccionadores (*routers*, *switches*)** Pueden hacer funciones de regeneración pero su función principal es direccionar los mensajes que le llegan, dirigiéndolos hacia un lado u otro según su destinatario. Se utilizan habitualmente para unir redes o subredes.

Subred

Sección de una red. Podéis ver el cableado estructurado en "Dimensionado".

- **Receptores**. Se denomina *receptor* a cada uno de los destinatarios a quien va dirigido el mensaje.

2.2. El modelo OSI

Toda red ha de realizar una serie de tareas para que pueda funcionar como red correctamente. El modelo OSI enumera estas tareas y las clasifica en siete niveles o capas.

El modelo OSI[3] es la referencia de todo estudio de redes. Se divide en siete capas o niveles que representan la división teórica de tareas que se recomienda efectuar para el estudio o diseño de cualquier red.

[3]OSI: *open system interconnection*

Cada capa tiene una función específica y las capas se apilan de manera que cada una depende de la inmediatamente inferior para que funcione correctamente.

Modelo OSI

No todas las redes han de cumplir exactamente con este modelo, de hecho, normalmente no lo hacen. De todas maneras, se recomienda tenerlo en cuenta como referencia, ya que su conocimiento facilita la comprensión de cualquier red y facilita poder hacer comparativas entre diferentes sistemas.

Las siete capas son las siguientes:

Capas del modelo de referencia OSI.

Las tareas de cada capa son las siguientes:

1) Física: define las reglas para transmitir el flujo de bits por el medio físico.

2) Enlace: organiza los bits en grupos lógicos denominados *tramas*, proporciona información para el control del flujo y para el control de errores.

3) Red: proporciona la posibilidad de direccionar la información agrupada en paquetes.

4) Transporte: garantiza que los datos llegan correctos del emisor al destinatario. Independientemente del medio de físico sobre el que viajan los datos.

5) Sesión: gestiona la conexión y el mantenimiento del enlace.

6) Presentación: a menudo forma parte del sistema operativo y se encarga de dar forma a los datos.

7) Aplicación: son los servicios destinados al usuario que se montan sobre la base de una red; por ejemplo, el servicio de correo electrónico.

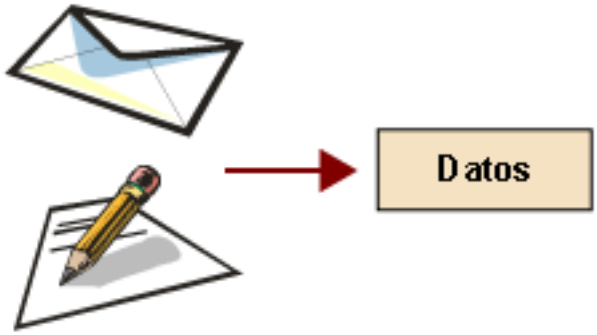
En las definiciones de las capas se ha hablado de tramas y paquetes, en el apartado de encapsulación se explica qué significan.

2.2.1. Encapsulación

Cuando un ordenador envía datos a otro por medio de la red, éstos no viajan solos, sino que lo hacen junto a mucha más información útil para verificar errores, dirección de destino, control de flujo, etc. Toda esta información se añade de una manera ordenada por medio de un procedimiento denominado *encapsulación*.

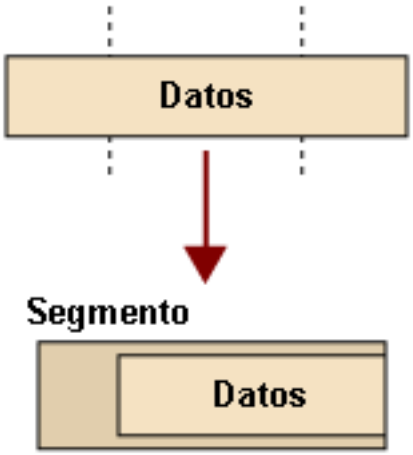
Cuando los datos se envían de un ordenador a otro, viajan mediante las capas del modelo OSI; las tres capas superiores (aplicación, presentación y sesión) preparan los datos para ser transmitidos y crean un formato común para la transmisión. Una vez tenemos el formato común, la encapsulación envuelve los datos con la información necesaria según los protocolos antes de que se una al tráfico de la red. Es decir, a medida que los datos van pasando por las capas del modelo OSI reciben cabeceras, información final y más tipos de información que, al fin, garantiza que los datos lleguen a su destino físico y que, una vez allí suban, desde la capa física hasta la capa de aplicación, hasta presentarse en pantalla, almacenados en una base de datos, etc.

En la capa de presentación se crean los datos; cuando un usuario escribe un correo para enviar por correo electrónico o quiere enviar un documento de texto, sus caracteres alfanuméricos se pasan a datos binarios que se pueden transmitir por la red.



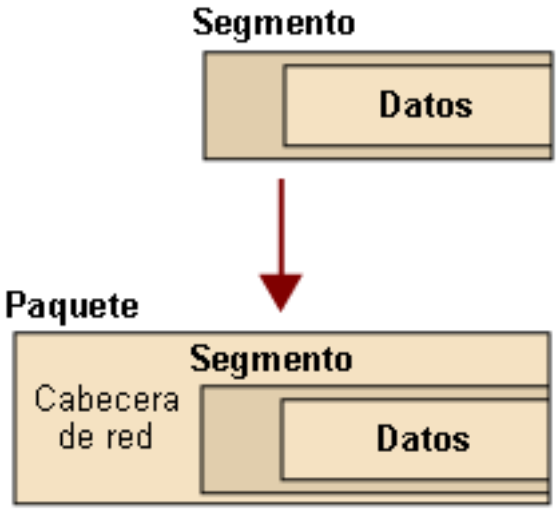
En la capa de presentación se transforma la información alfanumérica a datos binarios.

En la capa de transporte se dividen los datos en unidades más pequeñas que se pueden gestionar mejor, se dividen en segmentos de datos. A cada segmento se le da un número para que en el destino se puedan reordenar y se pueda recuperar el total de los datos en el orden correcto.



La capa de transporte divide el tren de datos binarios en segmentos.

En la capa de red se encapsula el segmento creando un paquete o datagrama. En el segmento se añade información como las direcciones lógicas de la máquina destinataria. Estas direcciones ayudan a los direccionadores a elegir la ruta que ha de seguir el paquete.

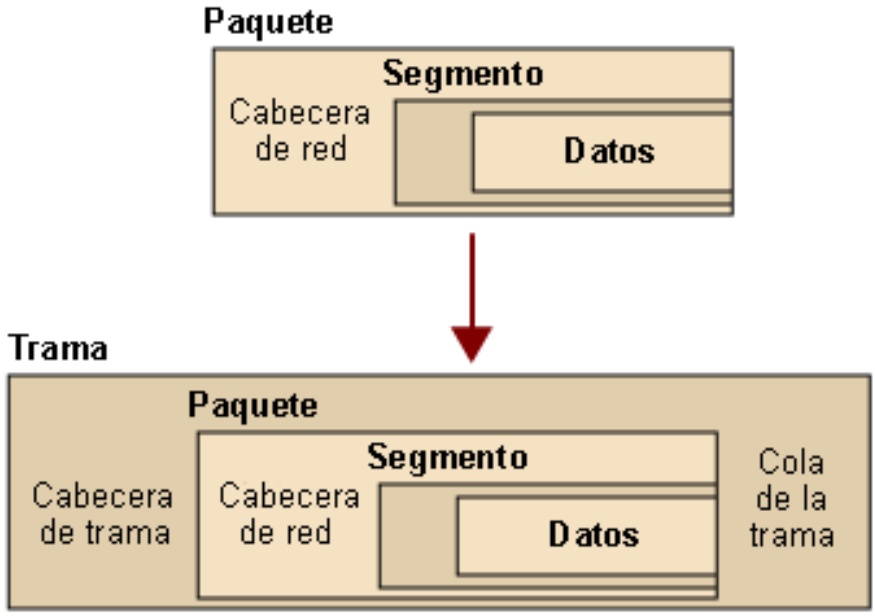


En la capa de red se encajan los segmentos en paquetes de datos.

En la capa de enlace continúa la encapsulación del paquete con la creación de una trama, a la cabecera de la trama se le añade la dirección MAC del origen y el destino. La capa de enlace transmite los datos en forma de bits para los medios de la capa física.

Dirección MAC

Número único e irrepetible que llevan todas las tarjetas de red.



Finalmente, en la capa de enlace se encajan los paquetes dentro de las tramas que viajarán por la red.

Finalmente, el tren de bits originado se transmite en la red por medios físicos (cableado, ondas, etc.).

Las tres capas inferiores del modelo OSI son las capas principales del transporte de datos por medio de una red interna o inherente.

2.3. Tipología de una red

Una de las maneras como podemos clasificar las redes es en función de su extensión.

De esta manera tenemos tres tipos básicos de redes: **WAN**, **MAN** y **LAN**.

WAN (*waste area network*, 'red de área extendida')	Son redes que dan servicio a grandes extensiones de territorio. Es una WAN la red de transporte de la señal telefónica o la red Internet.
MAN (*metropolitan area network*, 'red de área metropolitana')	Las redes MAN dan servicio a extensiones moderadamente grandes de territorio, típicamente son redes urbanas. Se considera una red MAN el cableado de un campus universitario, el de un polígono industrial o la red formada por las comunicaciones entre los distintos edificios de una entidad, siempre que estén dentro de un mismo ámbito urbano.
LAN (*local area network*, 'red de área local')	Las redes LAN dan servicio a pequeñas extensiones de terreno. Denominamos red LAN al cableado que ocupa una oficina, la planta de un edificio, un edificio entero, una escuela, el conjunto edificio de oficinas más almacén que tienen muchas empresas, etc.

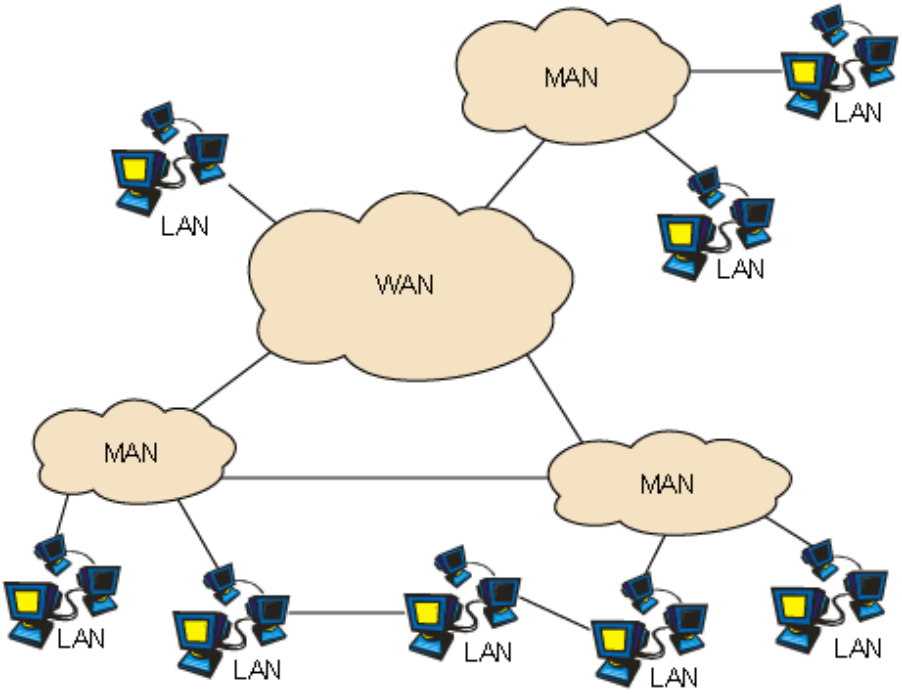
Para los objetivos del curso, de los tres tipos de redes, las que más nos interesan son las redes LAN, ya que son aquéllas con las que habitualmente se trabaja en el ámbito de usuario; las WAN y las MAN son redes que tienen su dominio en

el transporte de información a grandes distancias, que suele ser más del interés de las grandes operadoras de telecomunicaciones. Las redes que encontramos en oficinas, fábricas, almacenes, etc. forman siempre una LAN. Empresas, bibliotecas, archivos, centros médicos o ayuntamientos pueden tener una MAN/WAN que ponga en contacto las diferentes LAN que tienen en cada edificio.

Combinar los diferentes tipos de redes no es incompatible; al contrario, a menudo es recomendable o imprescindible.

Ejemplo de bibliotecas de universidad

Pongamos, por ejemplo, el conjunto de bibliotecas de una universidad; los terminales de los PC de la biblioteca de cada facultad estarán unidos en red mediante la LAN del edificio donde se encuentre, pero obviamente una de las ventajas que nos aporta la informática es compartir la información y sería imperdonable que las diferentes bibliotecas no compartieran información interna útil para su trabajo. La unión de la información de las diferentes bibliotecas formaría una MAN. Para acabar de tener un sistema completo, la universidad podría tener convenios con otras universidades de otras ciudades del país o del extranjero para compartir los datos de sus redes de bibliotecas, estos datos estarían unidos mediante una red WAN.



Modelo de posibles interrelaciones entre LAN, MAN y WAN.

En la figura anterior se puede ver que los tres tipos de redes se pueden unir entre ellas y que cualquier combinación es posible.

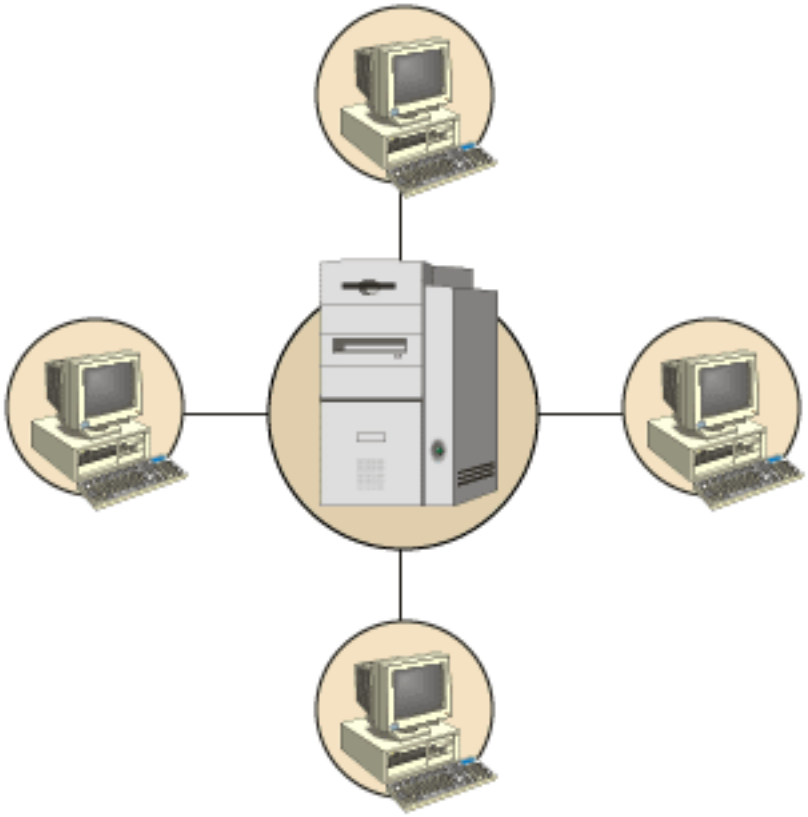
2.4. Topología de redes

Hay varias maneras de unir los componentes (ordenadores, impresoras, faxes, discos duros de red, etc.) que forman la red. Se pueden clasificar diferentes tipos de red según la topología del cableado de ésta. Para las redes LAN hay tres tipologías básicas.

2.4.1. Cableado en estrella

La conexión física del cableado se caracteriza por estar todas las estaciones de trabajo están unidas físicamente con una estación central, que es la encargada de gestionar las comunicaciones de la red.

Modelo de topología en estrella

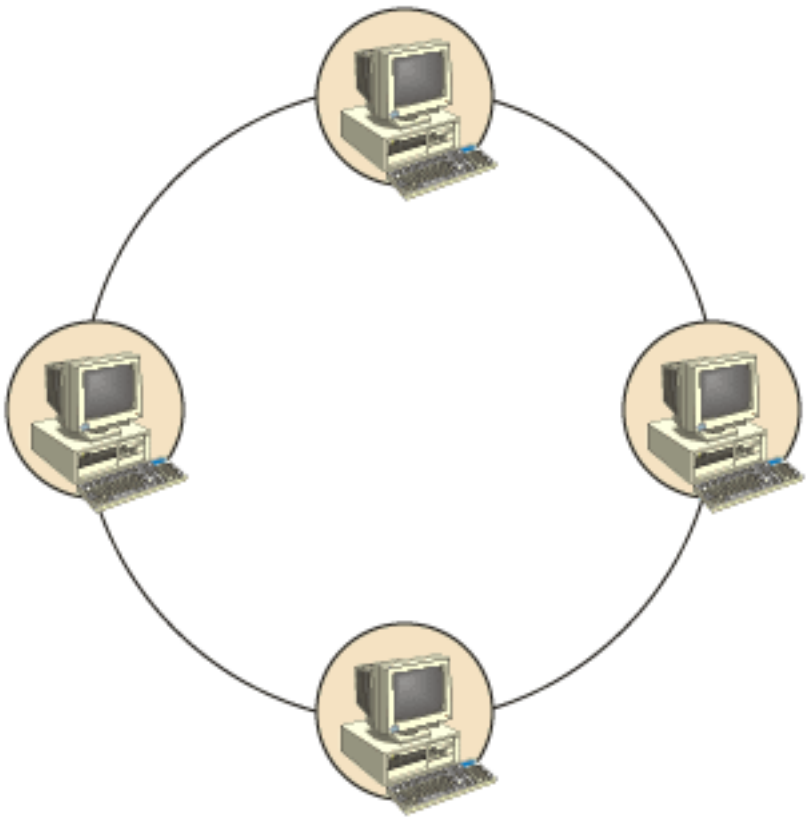


Ventajas	Inconvenientes
• Cada estación de trabajo puede utilizar una velocidad de comunicación diferente. La estación central será la encargada de gestionar el flujo de la información adecuadamente cuando se transmite entre estaciones que funcionan a diferentes velocidades.	• Si deja de funcionar el nodo central de la red, la red deja de funcionar, dicho en argot informático, cae la red. • Elevado tráfico en el nodo central de la red.
• Se detecta el origen de las averías fácilmente y el hecho de que una estación de trabajo deje de funcionar no afecta al funcionamiento de la red desde el punto de vista de las otras estaciones, a excepción de que la estación de trabajo que deje de funcionar sea la central.	• Todas las comunicaciones han de pasar por el nodo central de la red, lo que hace aparecer la posibilidad de que se produzca un cuello de botella si la máquina central no es lo bastante potente para atender todas las transmisiones a la velocidad adecuada; en este caso, la red se lentifica. Cuanto mayor sea el número de estaciones de trabajo en la red, mayor es el riesgo de sufrir este cuello de botella.

2.4.2. Cableado en anillo

Topológicamente, el cableado en anillo une todas las estaciones de trabajo formando un anillo, de manera que cada estación está unida a la que la precede y a la que le sucede, y así sucesivamente hasta completar un círculo lógico.

Modelo de topología en anillo

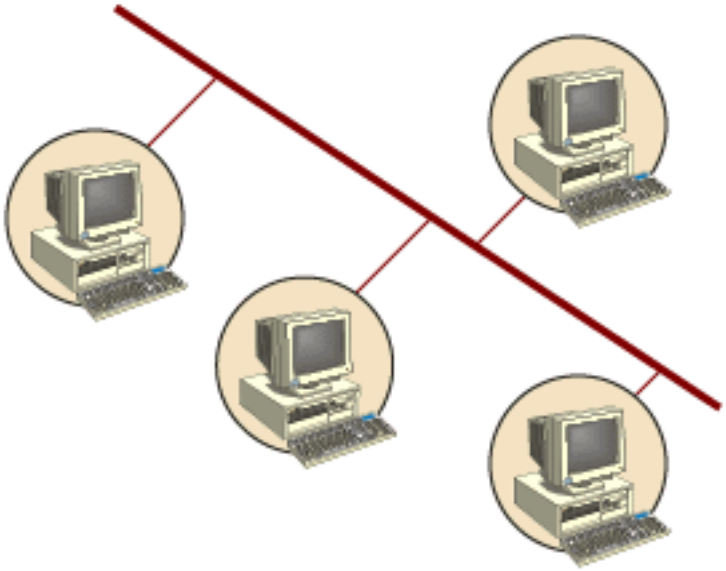


Ventajas	Inconvenientes
• La más destacada es que ofrece una tasa de errores muy baja, ya que las estaciones no retransmiten la información tal como les llega, sino que la leen, la interpretan y, en caso de no ser ellas las destinatarias, la regeneran antes de retransmitirla de nuevo a la red.	• Es una red frágil, si se desconecta una de las estaciones de trabajo, la red deja de funcionar, ya que se ha abierto el anillo y al hacerlo es como si hubiéramos cortado el medio sobre el cual circula la información. • Cuando hay un gran número de estaciones de trabajo puede ser una red lenta por su propia idiosincrasia, ya que los mensajes no van nunca por el camino más corto, sino que siguen un orden estricto por medio de las estaciones que forman el anillo. En cada estación se ha de leer la información que llega, consultar si es la estación destinataria o si no lo es, regenerar el mensaje y enviarlo a la estación siguiente de la red.

2.4.3. Cableado en bus

Para el caso de redes en forma de bus, el cableado se dispone de manera que hay un tronco central, un cable común, al cual se "pinchan" todas las estaciones de trabajo.

Modelo de topología en bus



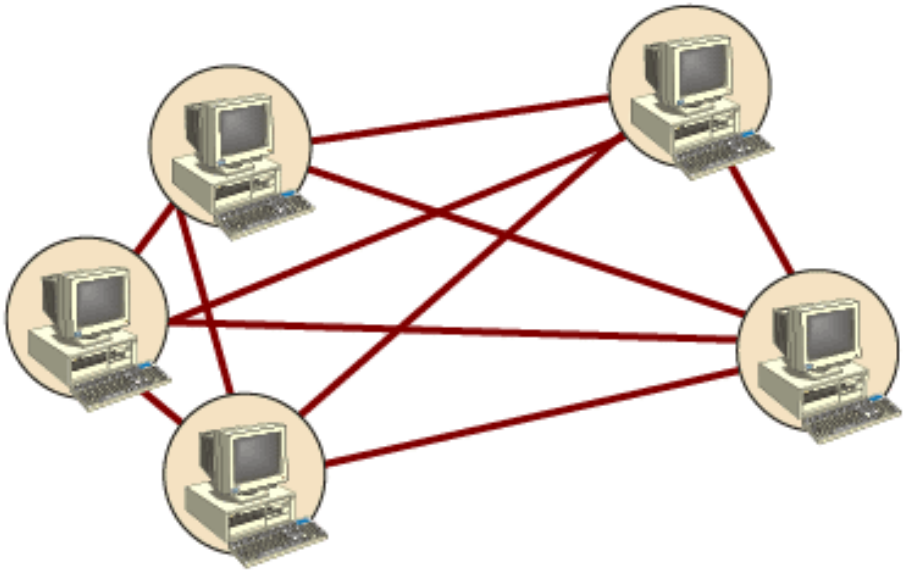
Ventajas	Inconvenientes
• Instalación muy simple y, en consecuencia, con un coste bajo. • Facilidad de adaptación de la topología a cualquier distribución espacial de las estaciones de trabajo en el edificio. • Es una arquitectura flexible en el dimensionado, es fácilmente ampliable y reducible.	• Un corte o mal funcionamiento del tronco central dejaría la red fuera de uso. • El bus debe estar bien dimensionado y tener suficiente capacidad para soportar toda la carga de trabajo que le exigirán todas las estaciones de trabajo. Si no es así, el mismo tronco central puede actuar de cuello de botella y lentificar la red.

2.4.4. Otras topologías

Se han presentado las tres topologías básicas de red, pero no son las únicas posibles.

Red mallada

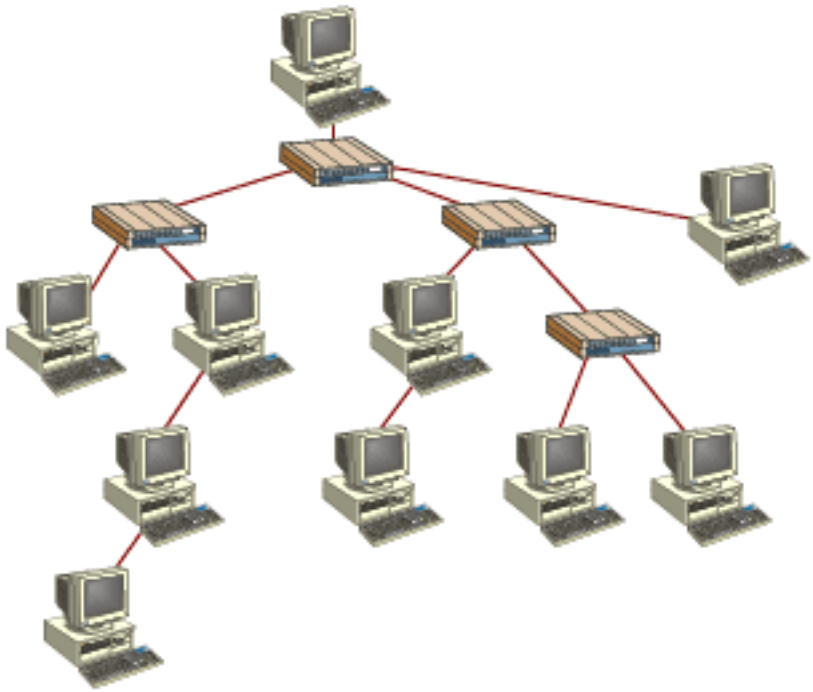
Es una topología más propia de redes WAN, se denomina *mallada* porque todas las estaciones se pueden comunicar unas con otras por diferentes rutas. Son redes muy robustas ya que, si deja de funcionar uno de los enlaces, siempre hay un camino alternativo para hacer llegar la información.



Modelo de topología en estrella.

Red jerárquica

En una red jerárquica, partiendo de un equipo central los elementos de la red se van ramificando. En el gráfico no todos los nodos de la red son estaciones de trabajo, de hecho, en una estructura como ésta a menudo en los nodos hay elementos de red como concentradores (*hubs*) que unen los diferentes elementos de la red mediante electrónica interna.

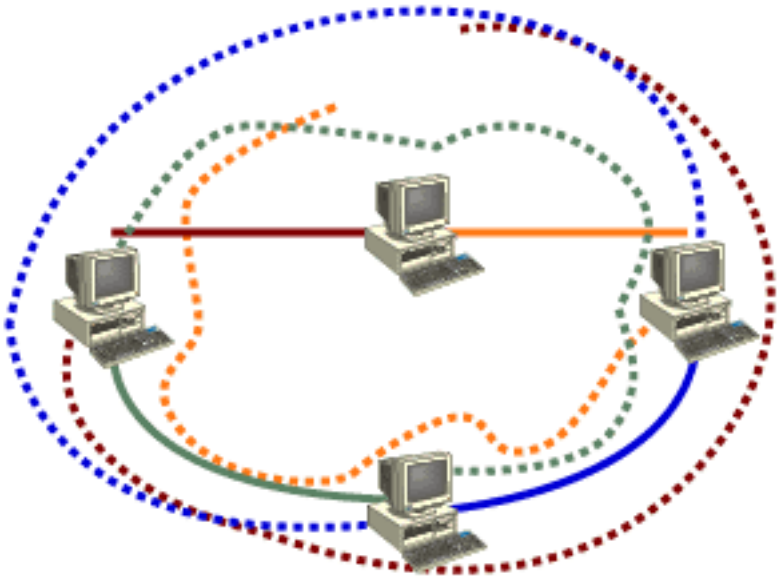


Modelo de topología en árbol (topología jerárquica).

Anillo mallado

En este caso hay un anillo doble, es decir, tenemos una redundancia, ya que aparte del anillo principal (líneas continuas), tenemos un segundo anillo (líneas discontinuas). Esta topología, a pesar de que redundante, es interesante

para un tipo de red como las MAN. Hay que observar que si accidentalmente alguien, como por ejemplo una excavadora de obras, corta el tubo por donde pasan los cables entre dos de los nodos de la red, no pasa nada si continúa estando el anillo intacto.



Modelo de topología en anillo mallado.

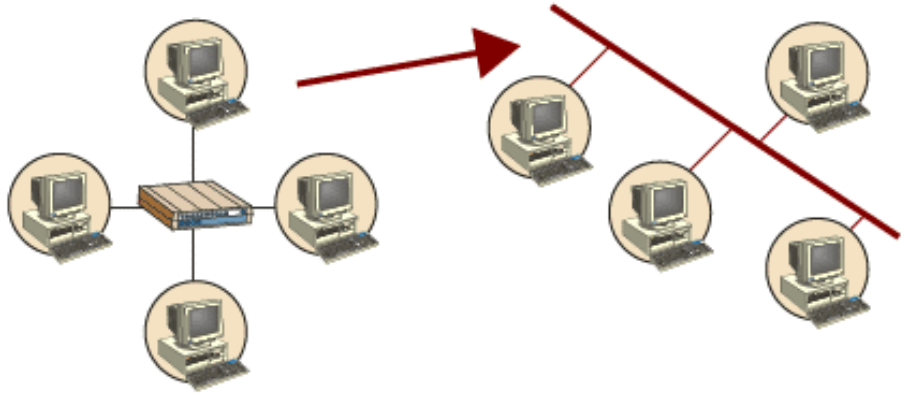
2.4.5. Topología física y topología lógica

Hay que saber distinguir entre la estructura física de la red, cómo se unen los elementos, y su estructura lógica, cómo se comportan.

Teniendo en cuenta este matiz, se puede implementar un cableado en estrella y que después se comporte como un cableado en bus o en anillo. Igualmente, puede hacer un cableado en forma de árbol y que después se comporte como un bus.

Diferenciar la topología física de la lógica no se hace para complicar más la situación, al contrario, se hace para facilitar el cableado. A menudo es más cómodo y fácil poner un concentrador y conectar las estaciones de trabajo de una sala.

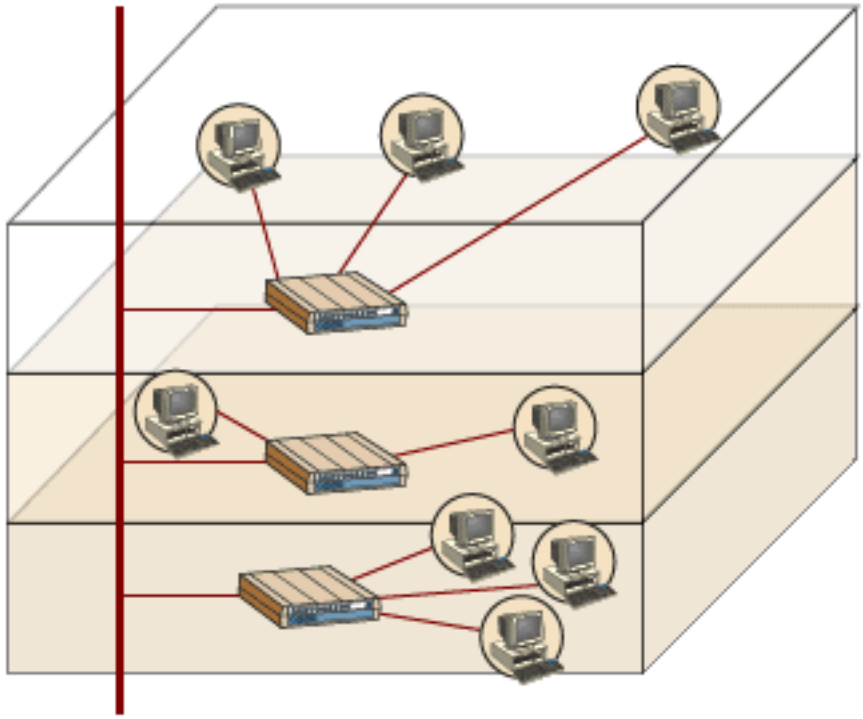
En la figura siguiente se puede ver que un cableado físico en estrella, en el que el nodo central es un concentrador (*hub*), es perfectamente convertible en un cableado lógico en bus. Gracias a las características del concentrador, está haciendo de tronco central del bus.



Topología física en estrella, topología lógica en bus.

Se pueden combinar las diferentes situaciones que se han visto.

En la figura siguiente se ve cómo se puede solucionar el cableado para una entidad que ocupa tres plantas de un edificio. Una posibilidad es poner un bus que recurra la vertical de los tres edificios, conectado al bus en cada piso puede haber un *hub* y conectado al *hub* todas las estaciones de trabajo, impresoras, etc. Tenemos que en cada planta hay una topología física en estrella, si lo miramos desde el punto de vista del tronco vertical, es una topología en bus. Todo el conjunto tendrá sentido si lo hacemos funcionar como una topología lógica en forma de bus.



Topología lógica de bus. Topología física, combinada con bus y estrella.

2.4.6. Cableado estructurado

Hasta ahora hemos visto que tipológicamente había unos modelos básicos, pero la topología física real de una red se puede llegar a complicar mucho más allá de estos modelos básicos. Si la red es muy grande, o las necesidades de

la empresa, institución, etc. lo requieren, es bueno dividir la red en secciones y unirlas mediante direccionadores. De esta manera, la red queda seccionada en pequeñas redes unidas entre ellas por los direccionadores. Esta solución se puede adaptar según varios criterios:

- **Eficiencia**: si la red es muy grande, y en función del protocolo que pongamos, probablemente será más rápida si está seccionada.

- **Técnicos**: si el número de estaciones es muy elevado, nos podríamos quedar sin bastantes direcciones IP en función del tipo de red que tengamos (recordemos que hay casos en los que el identificador de estación de trabajo era un único byte, es decir, como mucho 256 estaciones de trabajo). Al seccionarla, en la dirección IP cada subred tendría un identificador de red diferente, de manera que aunque repitiéramos un identificador del nodo, la dirección IP ya sería diferente (**192.168.1**.1, **192.168.2**.1).

- **Seguridad**: para mantener aislada una parte de la información de la red respecto del resto. Seccionando esta parte de la red, es más fácil mantener el control mediante el direccionador que lo une al resto.

- **De organización y de mantenimiento**: tener la red seccionada ayuda a tenerla organizada, a detectar averías y a llevar a cabo su mantenimiento.

2.5. Protocolos de acceso a la red

Las redes tienen por objetivo el intercambio de información entre los elementos que la componen (estaciones de trabajo, impresoras, discos duros compartidos, etc.).

Este intercambio no se puede realizar de una manera anárquica, es necesario que el acceso a la red esté regulado.

Si todos los integrantes introducen información a la red en el momento en el que les apetece, se producirían colisiones de la información, en otras palabras, los mensajes se superpondrían unos a otros y se producirían interferencias, como sucede en un debate entre un grupo de personas cuando no hay orden: si todas hablan a la vez, es difícil que se entiendan. Para evitar este caos, un debate requiere un moderador que gestione los turnos de palabra y que los haga respetar. En las redes, esta figura del moderador la asume el protocolo de acceso a la red.

Hay muchos protocolos de acceso a red pero explicaremos los más utilizados en redes LAN según su topología.

2.5.1. CSMA/CD (*carrier sense multiple access/collision detection*)

El protocolo CSMA/CD (acceso múltiple por observación de la portadora con detección de colisión) se implementa en topologías de bus, típicamente la velocidad del bus es de 10 Mbit/seg o de 100 Mbit/seg. El cableado se suele realizar con hilo coaxial, aunque para redes reducidas también se puede hacer con hilos de par trenzado.

Cualquiera de las estaciones de la red, antes de emitir un mensaje, debe respetar los pasos indicados por el protocolo:

1) Antes de enviar algo a la red es necesario que mire si ya hay alguien que la utiliza, algo que denominamos *escuchar la red*. Si no oímos nada (únicamente la señal portadora), entonces podemos acceder a transmitir información; en caso contrario, hemos de esperar a que acabe la transmisión en la red.

2) El mensaje llegará a todas las estaciones de trabajo por medio del bus que conforma la red, cada una consultará la cabecera (en la mayoría de los protocolos, dentro de la cabecera es donde se encuentra la información de los destinatarios) del mensaje y, si está entre los destinatarios, se copiará el mensaje.

3) Si durante el proceso de envío del mensaje aparece otra estación de trabajo que desea enviar alguna información, deberá proceder como hemos explicado. Durante la fase de escucha, al ver que la red está en uso, esperará un tiempo prudencial (normalmente es un tiempo aleatorio, del orden de fracciones de segundo) y volverá a realizar la escucha, y así sucesivamente hasta que la escucha dé como única señal la señal portadora, momento en el que la red está libre y procederá a emitir el mensaje.

4) Si en los primeros instantes del inicio de la transmisión de un mensaje otra estación también quiere transmitir su información, puede suceder que no detecte que el bus está ocupado porque el mensaje todavía no le ha llegado (éste es un caso posible, aunque muy improbable, dado que los mensajes viajan por el bus a grandes velocidades y el tiempo que pasa entre que se emite el mensaje y que llega a las estaciones es mínimo). Suponiendo que no se detecte que el bus está ocupado cuando esta segunda estación transmita, se producirá una colisión entre los mensajes, las dos estaciones de trabajo detectarán la colisión e interrumpirán las emisiones respectivas. Esperarán un tiempo aleatorio y volverán a empezar el proceso de emisión siguiendo de nuevo todos los pasos del protocolo. Las estaciones involucradas en una colisión pueden ser más de dos, pero, en cualquier caso, el procedimiento que se ha de seguir es el que hemos descrito.

2.5.2. Anillo de testigo (*token ring*)

El protocolo *token ring* ('anillo con paso de testigo') se implementa en topologías de anillo. Típicamente, la velocidad de la red oscila entre los 4 Mbit/seg y los 16 Mbit/seg. El cableado del anillo suele ser hilo coaxial, aunque también es habitual encontrar anillos con hilo de pares trenzados.

El acceso a la red se gestiona de una manera tan curiosa como eficaz, aunque la eficiencia en el uso de la red es discutible. La clave es saber que hay un pequeño mensaje denominado *anillo* (*token*) o *testigo* que no para de dar vueltas al anillo.

Las diferentes estaciones de trabajo retransmiten la marca siguiendo un estricto orden, de manera que el testigo recorre el anillo indefinidamente. Poseer el testigo da derecho a hacer uso de la red, a transmitir información.

El comportamiento general de este protocolo de acceso se puede describir de esta manera:

1) Cuando una estación recibe el anillo puede ser que tenga algo que transmitir o no. En caso de que no tenga nada que transmitir, transmitirá el anillo hacia la estación del anillo siguiente. En caso de tener algún mensaje para transmitir, retirará el testigo del anillo y transmitirá el mensaje.

2) Una vez transmitido, el mensaje viajará por cada una de las estaciones hasta que vuelva a la estación que lo ha enviado. Las estaciones leen el destinatario del mensaje y, en caso de que sean ellas, lo copian; en cualquier caso, lo vuelven a enviar hacia la estación de trabajo siguiente.

3) Finalmente, el mensaje transmitido llega a la estación emisora y ésta lo saca de la red. Como el anillo continúa en su poder, si tiene otros mensajes que enviar repetirá el proceso. En caso contrario, enviará el anillo a la estación del anillo siguiente para que pueda, si lo requiere, tomar el anillo y empezar la transmisión de mensajes.

2.5.3. Bus de testigo (*token bus*)

El protocolo *token bus* ('bus con paso de testigo') se implementa en topologías de bus. Igual que en el caso de redes CSMA/CD, el cable puede ser tanto coaxial como par trenzado.

El acceso a la red se gestiona mediante el paso de testigo, al igual que en el caso del protocolo anillo de testigo. Como se puede ver, es una mezcla de los dos casos anteriores, tenemos una topología en bus y un protocolo basado en un paso de testigo. La pregunta parece evidente: ¿cómo se puede implementar un paso de testigo si no tenemos una topología en anillo? ¿A quién le pasará el testigo la última estación de bus?

La respuesta a las preguntas anteriores es la creación de un anillo lógico sobre el bus, es decir, las estaciones de trabajo están numeradas, de manera que cada estación debe transmitir el testigo a la estación con el número superior y la de número mayor lo transmite a la de número menor. De esta manera, conseguimos un anillo lógico sobre un cableado en bus. A partir de aquí, la operativa se explica en el anillo de testigo.

El anillo de testigo se suele implementar en entornos ofimáticos, donde la creación física de un anillo no suele presentar muchos problemas. El bus de testigo se utiliza en entornos industriales, ya que es más fácil y en general más económico hacer un cableado en forma de bus.

El bus de testigo tiene la robustez del anillo de testigo y, además, se aprovecha de la facilidad de conectar y desconectar estaciones de trabajo que nos ofrecen los cableados en bus.

2.6. Redes sin hilos o WIFI

Una red WIFI o sin hilos actúa de manera idéntica a una red con un cableado con hilo coaxial, par trenzado, fibra, etc.

Si bien funcionalmente es idéntica, varía el hardware del "cableado". El medio de transmisión es el aire y, para depositar la información en el aire mediante ondas electromagnéticas, serán necesarias antenas transmisoras y receptoras para volver a capturarlas.

La velocidad actual de los sistemas WIFI actuales puede llegar a los 100 Mbps. En la actualidad la velocidad más extendida en sistemas WIFI es de 54 Mbps. Para aquellos elementos que cumplen la normativa 802.11g. 54 Mbps es una velocidad suficiente para que una red LAN funcione sin problemas. Los elementos de hardware que cumplen la norma 802.11n son los que nos permiten llegar a velocidades superiores a los 100 Mbps; no hay bastante con tener alguna antena que cumpla esta norma para coger estas velocidades, ya que la velocidad de una red siempre es la velocidad de su elemento más lento.

Es posible que aparezca la necesidad de poner antenas repetidoras por las causas siguientes:

- **La distancia**: cuando la distancia entre ordenadores es muy grande. Sabemos que la distancia es grande o no leyendo las especificaciones del punto de acceso, ya que cada punto de acceso emite una potencia determinada.

- **Obstáculos**: si hay obstáculos que atenúen mucho la señal, como pueden ser paredes gruesas o de hormigón.

Actualmente, en cualquier tienda especializada se dispone de una amplia gama de elementos para crear y aportar valor añadido a una red WIFI.

- Tarjetas de red.

Tarjeta WIFI para *slot* PCI (para ordenador de sobremesa)

- Tarjetas de red PCMIA.

Tarjeta WIFI para *slot* PCMIA (para ordenador portátil)

- Puntos de acceso (*access point*).

Puntos de acceso WIFI

- Antenas/repetidores (*air points*).

Antenas WIFI

- Direccionadores (*routers, access point*).

Direccionador con soporte para WIFI

Si queremos unir dos ordenadores mediante un enlace WIFI, tenemos suficiente con equipar los ordenadores con tarjetas de red WIFI.

Si queremos unir más de dos ordenadores, nos hará falta un punto concentrador (*hub*) o direccionador preparado para red WIFI.

A menudo las redes son mixtas, de manera que en la misma LAN hay ordenadores unidos por un cableado clásico y otros mediante WIFI.

Pongamos un ejemplo habitual: una vivienda en la que se dispone de un acceso ADSL, un ordenador de sobremesa (ordenador 1) en una habitación con el acceso ADSL, una impresora, un segundo ordenador (ordenador 2) en otra habitación y un ordenador portátil. Para no tener que romper la estética de las paredes con cables, ni tener que hacer regatas en las paredes, una solución es unir los tres ordenadores y la impresora mediante una red WIFI. El objetivo es conseguir que los ordenadores puedan compartir datos y que los tres aprovechen el acceso ADSL de Internet.

¿Qué nos hará falta?

- Direccionador ADSL con WIFI: se conectará a la roseta telefónica por donde entra la señal ADSL y la tarjeta de red.

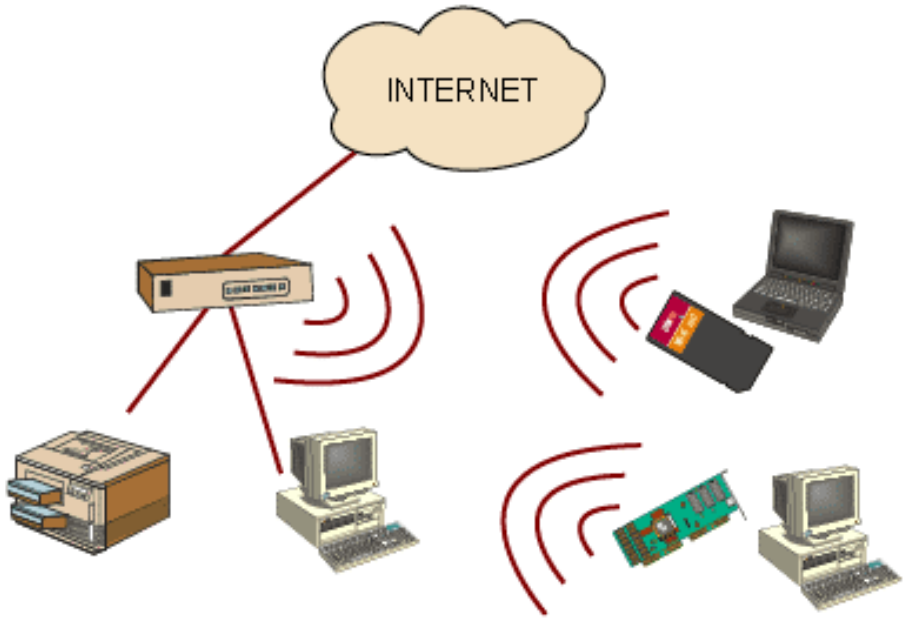
- Tarjeta de red: se instalará en el ordenador 1 de sobremesa de la habitación donde está la roseta telefónica con ADSL.

- Tarjeta de red WIFI: se deberá poner en el ordenador 2 de sobremesa.

- Tarjeta de red PCMIA: se pone en la ranura PCMIA del portátil. Este elemento es probablemente innecesario, pues la mayoría de ordenadores portátiles actuales disponen de serie de antena WIFI.

- Cable: del ordenador 1 al direccionador WIFI y de la impresora al direccionador WIFI. En caso de que la impresora no se pueda conectar en red,

siempre se podría conectar al ordenador 1 y mediante la configuración adecuada desde el sistema operativo, los otros ordenadores la verían como si fuera una impresora en red. Otra opción es conectar la impresora a un dispositivo servidor de impresión WIFI y, de esta manera, convertimos la impresora en una impresora en red accesible para todos los ordenadores.



Modelo WIFI para una vivienda con dos ordenadores de sobremesa, un portátil e impresora.

En la figura anterior vemos la topología de la red. Tanto el ordenador 1 como la impresora también podrían estar conectados al direccionador por la red WIFI, con lo que se evitaría poner ni siquiera un metro de cable, el ordenador mediante una tarjeta WIFI como la que se dispone en el ordenador 2 y la impresora mediante un punto de acceso.

2.7. Valores añadidos de la Red en el hogar doméstico

La evolución de los aparatos electrónicos dedicados al hogar ha hecho que éstos estén cada vez más preparados para conectarse a las redes domésticas. Así, no es difícil encontrar televisores con entradas por red DLNA, aparatos NAS para tener discos duros conectados a la Red, cadenas de alta fidelidad con opciones de *streaming* de manera que, desde cualquiera de estos aparatos, se puede acceder a la información de la Red y reproducir en la televisión vídeo, imagen y sonido en los aparatos que tengan opciones de *streaming*.

Algunos de los elementos que se pueden encontrar con más facilidad, para conectar a la WIFI y dar un valor añadido, son:

- **Adaptador para disco duro en red**. Permiten conectar discos duros e integrarlos en la Red, de modo que su contenido se pueda compartir.

Adaptador para disco duro en red.

- **Servidor de impresión WIFI**. Convierten cualquier impresora en una impresora accesible para todos los dispositivos de la Red.

Servidor de impresión WIFI.

- **Webcam WIFI**. Este dispositivo permite ver imágenes del lugar en el que esté ubicado, desde cualquier ordenador conectado a Internet. Obviamente, hay protocolos de seguridad que hacen que se pueda restringir el acceso a las imágenes. Son también utilizados como cámaras de seguridad.

Webcam WIFI.

- **Disco duro en red NAS**. Este dispositivo permite compartir ficheros con mucha facilidad; es ideal para gestionar las copias de seguridad y da unas posibilidades enormes de expansión de la capacidad de almacenaje de datos en la Red.

Disco duro en red NAS.

- **Teléfono IP por red WIFI**. La telefonía IP permite usar la infraestructura de la Red para comunicar vía voz, telefónicamente, con cualquier otro teléfono IP.

Teléfono IP por red WIFI.

- **TV con conexión LDNA**. La conexión LDNA nos permite reproducir por la televisión los contenidos almacenados en los dispositivos de la Red, bien sean los discos duros de los ordenadores, los discos duros de la Red o bien dispositivos USB, tarjetas Compact Flash, tarjetas SD, etc.

TV con conexión LDNA.

- **Consola con conexión WIFI**. Las consolas de videojuegos como la Wii permiten la navegación por Internet, compartir datos de los juegos e incluso jugar en línea (en tiempo real) con personas de todo el mundo.

Consola con conexión WIFI.

3. Internet e intranet

Todos los estudiantes de la UOC conocen Internet y su potencial. Ahora nos interesa ver Internet como una red, desde el punto de vista del responsable de gestión de información. Internet es, aparte de una fuente de información inacabable, una herramienta de comunicación entre fuentes de datos situadas, quizá, a miles de kilómetros unas de otras.

Por este motivo, la unidad empieza preguntando qué es Internet como red y encontraremos la respuesta a lo largo de la unidad. Para responder a la pregunta hay que bajar a la simple descripción para ver cómo se fundamenta Internet técnicamente. Igual que en las otras unidades, estas incursiones a bajo nivel, de carácter técnico, sólo pretenden fundamentar bien los conocimientos descriptivos y conceptuales, que son el objetivo real que se busca.

Se explican los protocolos que dan soporte a la red Internet y permiten que crezca sin problemas día tras día, se explican los servicios que hay construidos sobre este soporte y cuáles son los protocolos de cada uno y, finalmente, se hace una referencia a las aplicaciones web, que apuntan que serán el futuro en la ingeniería del software, ya que cada vez son más, funcionan mejor y, por consiguiente, están más extendidos.

3.1. Internet

Internet ya no es el futuro, sino el presente. Incluso a mucha gente le sería difícil desarrollar su tarea profesional sin usar esta herramienta, o mejor dicho, sin usar los servicios que la red Internet ofrece.

Internet está presente en nuestras vidas; cuando se usa el correo electrónico, cuando se estudia en la UOC, cuando se busca información, cuando se consultan cuentas bancarias desde la página web de la entidad correspondiente, cuando se usa para el ocio, etc. Hasta han aparecido nuevas patologías, como la adicción a Internet o a alguno de los servicios que ofrece. Aunque usar Internet sea una tarea cotidiana y, para algunos, imprescindible, no ha sido siempre así; las generaciones actuales no pueden percibir el mundo sin Internet, MSN, Facebook, el correo electrónico, los juegos en línea, etc.; todo esto forma parte cotidiana de su vida, pero cuando estos jóvenes y adolescentes nacieron probablemente sus padres no se habían conectado nunca a Internet. Ellos han crecido con la Red, y, en paralelo, la sociedad ha cambiado los hábitos de ocio, laborales e incluso de relación personal por influencia de la Red. Podemos decir, pues, que Internet ha tenido una evolución y popularización extraordinarias.

3.1.1. ¿Qué es Internet?

Esta pregunta tiene una respuesta difícil. Si alguien busca la respuesta, es probable que tropiece a menudo con definiciones del estilo "Internet es la red de redes". De hecho, ésta es la respuesta correcta. Aun siendo correcta, la respuesta deja indiferentes a la mayoría de las personas que buscaban una explicación de Internet. A lo largo de la unidad se intentará revelar qué hay detrás de esta frase.

Para empezar, podemos matizar la definición de Internet como red de redes diciendo que Internet es una red de extensión mundial formada por millones de ordenadores conectados unos con otros mediante equipos de comunicación (direccionadores, pasarelas o *gateways*, conmutadores o *switches*, proveedores de Internet, etc.). Todos con el denominador común del protocolo TCP/IP[4].

[4]TCP/IP: *transmission control protocol/Internet protocol*, explicados en el apartado "TCP/IP".

La existencia de este protocolo común hace que cualquier usuario de una de las redes que forman parte de Internet se pueda comunicar con un ordenador remoto que pertenece a otra de las redes que forman parte de Internet, de manera que si lo miramos desde una perspectiva de más alto nivel, los dos ordenadores forman parte de una única red, Internet.

3.1.2. Los orígenes de Internet

En los años sesenta, el Departamento de Defensa de Estados Unidos crea la Agencia de Investigación para Proyectos Adelantados (ARPA), dedicada a la investigación y con el objetivo de implementar la más alta tecnología a proyectos de carácter militar. Una de las obras de ARPA es crear la red ARPANET para resolver las comunicaciones del ejército.

En 1969, ARPA junto con la compañía Rand Corporation desarrollan una red sin nodos centrales: la información se divide en paquetes numerados que contenían la dirección del ordenador destino; en el destino, el ordenador receptor los ordenaba según la numeración y si faltaba alguno solicitaba la retransmisión. Todo esto lo gestionaba un protocolo denominado NCP (*network control protocol*).

En 1974, los científicos de ARPA, junto con expertos de la Universidad de Stanford, crean las actuales bases de Internet, modifican el protocolo NCP para dar a luz el protocolo TCP/IP.

En 1982 se adopta el protocolo TCP/IP como el estándar de comunicación para redes conectadas a ARPANET, que ya ha dejado de ser una red militar para pasar a tener finalidades civiles, sobre todo científicas.

ARPANET ha evolucionado mucho desde sus orígenes, pero la filosofía continúa siendo la de una red sin nodos, fácilmente escalable, donde todos los ordenadores se puedan conectar de una manera fácil. Cuando decimos que Internet es una red sin nodos queremos decir que su filosofía parte de que no hay jerarquías: todos los ordenadores conectados tienen el mismo nivel e importancia, todos se pueden conectar el uno con el otro. Desde la primera aplicación militar, ha tenido mejoras técnicas como el TCP/IP y su uso ya es con finalidades plenamente civiles. A partir de ahora, se habla de Internet y no de ARPANET, aunque no podemos dar una fecha concreta del cambio.

Nodo

En general, un nodo es el punto de unión entre varias redes. En Internet, un nodo es un *host* (servidor) con un solo nombre de dominio y dirección.

La gran revolución de Internet se produce cuando sobre el protocolo TCP/IP se construyen servicios como el correo electrónico o el World Wide Web (WWW) que aparece a principios de los años ochenta.

Poco a poco, servicios tradicionales como la radio, la banca, la Administración pública, la Universidad o la propia telefonía se van integrando en mayor o menor medida en la red de redes; todo hace que Internet crezca exponencialmente día a día.

A partir de estos momentos, Internet empieza a tener problemas; víctima de su éxito, la Red supera todas las previsiones de uso y su crecimiento es imparable hasta llegar a nuestros días, donde en el denominado primer mundo, todo el mundo tiene acceso a ella.

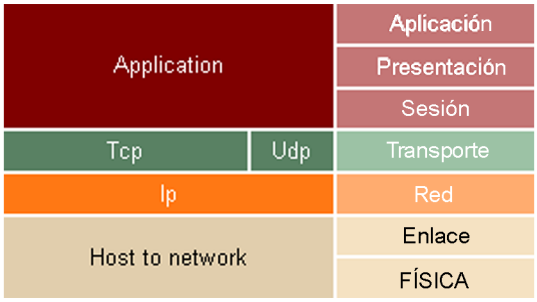
3.1.3. Protocolo TCP/IP

Aunque el objetivo del curso es la comprensión conceptual y no la profundización en la parte técnica, parece imprescindible detallar mínimamente qué se entiende por protocolo TCP/IP, ya que, como se ha dicho, Internet y sus servicios se construyen sobre esta base. Haciendo un símil, si Internet fuera un edificio, el protocolo TCP/IP sería los cimientos, y los servicios como el correo electrónico, WWW, FTP, etc., estarían en las plantas superiores del edificio.

El TCP/IP nos permite enlazar ordenadores que utilizan sistemas operativos diferentes. Esta característica es de gran importancia, ya que ordenadores con sistemas operativos diferentes en principio son dos mundos aislados. Con el TCP/IP no nos preocupa si accedemos a un ordenador que funciona con Linux, Unix o si es un Macintosh (Apple).

TCP/IP no es un único protocolo, está formado por el TCP (*transfer control protocol*) y el IP (*Internet protocol*). Al mismo tiempo, estos dos protocolos forman un conjunto estratificado en varias capas que dan lugar a lo que se denomina STACK de protocolos TCP/IP.

TCP/IP sólo posee cuatro niveles, con estos cuatro niveles cubre todas las capas de la recomendación OSI. En el gráfico siguiente, vemos los cuatro niveles de TCP/IP y las equivalencias con los niveles de la recomendación OSI.



Equivalencia de las capas del modelo OSI con las cuales se utiliza el protocolo TCP/IP.

IP (*Internet protocol* o protocolo de Internet)

Todo elemento conectado a una red TCP/IP debe tener una dirección IP que lo defina unívocamente.

Una dirección IP es un grupo de 4 bytes (32 bits); normalmente se expresan las direcciones IP en forma decimal separando cada byte mediante un punto, por ejemplo (192.168.1.2).

La dirección IP no es un número aleatorio. No es el objetivo del curso saber la reglamentación de las direcciones IP, pero sí que hay que saber que la dirección IP contiene mucha información, ya que identifica la red a la cual pertenece el elemento y la dirección de este elemento en la red. En función del tipo de red, se identifica por el primero, primero y segundo, o primero, segundo y tercer byte. En consecuencia, la dirección del elemento en la red identifica el cuarto, tercero y cuarto o bien los últimos tres bytes.

Cuando un ordenador pasa a ser un nodo de acceso a la misma red Internet, debe tener una dirección IP que le tiene que haber dado la Internet Assigned Number Authority (IANA), que es la entidad única a escala mundial encargada de asignar direcciones IP.

Una vez se posee una NetID (una dirección de red única), se puede conectar un direccionador, conmutador, etc. al ordenador y crear una LAN donde todos los ordenadores tengan acceso a Internet mediante la IP única que da salida a Internet (*gateway*).

Pasarela (*gateway*)

La pasarela es el elemento de la red que da salida a Internet.

Dentro de una LAN, es el administrador de la LAN quien decide cómo repartir las direcciones IP, siempre respetando la normativa.

No puede haber dos dispositivos dentro de una misma red con la misma IP. Elementos como *bridge* o *routers* son imprescindibles que conozcan las IP de las diferentes redes que enlazan. Una estación de trabajo puede tener la IP de su red LAN y, además, un módem que lo conecte directamente a Internet con la

IP que le haya asignado su proveedor. Es el caso habitual de los accesos ADSL domésticos; visto desde la LAN, el módem tendrá una IP dentro del rango de la red doméstica 192.168.X.Y; el módem tendrá, además, una IP pública, que es la que le da el proveedor de Internet. Esta IP pública puede ser dinámica o fija; si es dinámica, el proveedor le asigna en el momento que el *router* se pone en marcha, si es fija se le asigna en el momento de formalizar el contrato y se mantiene siempre igual.

IP es un protocolo de tercera capa OSI, identifica los elementos en la red pero se desentiende de cómo transportar la información. Es necesario que alguien se haga cargo de esta parte, ya que no sirve de nada saber perfectamente dónde hemos de enviar nuestro mensaje si no hay nadie que se encargue de hacer el trayecto hacia el destinatario. De ello se encarga el TCP.

TCP (*transmission control protocol*)

TCP es un protocolo de nivel 4 (transporte) que se encarga de garantizar que los mensajes llegan a su destinatario y que, además, llegan correctamente, es decir, sin errores.

La mayoría de los protocolos de transmisión tienen mecanismos para detectar que la información que llega sea fiel a la que ha emitido el emisor. Todos los mecanismos castigan la eficiencia del sistema, ya que hay que emitir información inútil desde el punto de vista del mensaje que se va a transmitir, pero que es imprescindible si se quiere realizar el control de errores. Si detecta que ha habido algún error, hace repetir el envío.

Control de errores

El control de errores puede llegar a ser muy complejo y fiable. Explicaremos uno de los controles de errores más básicos y usados, el bit de paridad.

Supongamos que el mensaje para transmitir es la siguiente cadena de 10 bits 1000110110; en la cadena, se añadiría el bit número 11, que hará las funciones de bit de paridad, y valdrá:

- 0 si el número de unos, dentro de los diez primeros bits, fuera par.
- 1 si el número de unos, dentro de los diez primeros bits, fuera impar.

Para el caso de transmitir 1000110110, el bit de paridad valdría 1 y se transmitiría 10001101101.

Para el caso de transmitir 1010110110, el bit de paridad valdría 0 y se transmitiría 10101101100.

Este es un control de errores muy poco robusto, ya que no detecta si hay un número par de errores. Como se puede ver en el ejemplo, poner un bit de paridad castiga la eficiencia del sistema; de los 11 bits transmitidos, sólo 10 llevan información; hemos perdido el 9,09% de la capacidad del canal.

El TCP utiliza metodologías de control de error más sofisticadas.

El TCP va unido indivisiblemente al concepto de puerto; un puerto es la unidad lógica que define el extremo de una comunicación. El TCP está orientado a la conexión, siempre va de un puerto origen a un puerto destinatario.

UDP (*user datagram protocol*)

UDP son siglas que podemos encontrar con facilidad en los textos que hablen de redes de comunicaciones en Internet. Es otro protocolo de nivel 4, encargado de la transmisión. Técnicamente, es mucho más simple que el TCP. Se apoya en IP y usa direcciones IP para transportar la información. Como el TCP, se encarga de garantizar que la información que llegue al destinatario sea fiable. UDP no es un protocolo orientado a la conexión y permite que el mensaje llegue a diferentes usuarios.

3.1.4. Aplicaciones y servicios

Hasta ahora hemos visto los orígenes, la filosofía y los fundamentos de Internet. Pero para que la Red sea realmente útil, es necesario que sobre este protocolo TCP/IP se construya alguna cosa que nos permita hacer algo de utilidad. Esta cosa son los servicios o aplicaciones en Internet, de muchas de las cuales somos usuarios habituales: correo electrónico, FTP, World Wide Web, Telnet, IRC, etc. Los servicios ocupan las capas de sesión, presentación y aplicación del modelo OSI. Cada servicio utiliza su propio protocolo que utilizará el TCP/IP para transmitir datos por la red. Algunos servicios son:

- **World Wide Web**: son las populares páginas web. Si se busca bien, podemos encontrar casi cualquier tipo de información en ellas. El servicio se comunica con el protocolo TCP/IP mediante el protocolo que utiliza el HTTP (*hipertext transfer protocol*). Hay una variedad del HTTP, el HTTPS (*hipertext transfer protocol security*), que la utilizan las páginas con información confidencial con el fin de ocultar datos a posibles escuchas malévolas.

- **Transmisión de archivos**: se utiliza para descargar archivos desde ordenadores remotos. El protocolo que se utiliza con esta finalidad es el FTP (*file transfer protocol*). También se pueden bajar archivos mediante el HTTP, pero es mucho más lento. En muchas páginas web, a la hora de descargar el archivo nos deja elegir si lo queremos hacer por HTTP o bien por FTP. La elección lógica es hacerlo por FTP, pero a menudo en las LAN el ordenador que da conexión a Internet (el *gateway*) tiene el protocolo FTP en capas para las estaciones de trabajo de la LAN. Éste es un criterio que adoptan muchos administradores de red por motivos de seguridad.

- **Correo electrónico**: los conocidos y sobradamente utilizados servicios de correo electrónico (*e-mail*), mediante éstos son capaces de recibir correo. El protocolo utilizado es el SMTP (*simple mail transfer protocol*). ¿Qué sucede cuando enviamos un correo desde una aplicación web, como el campus de la UOC, que también utiliza SMTP o bien que va sobre HTTP, que es el protocolo propio de la aplicación web? La aplicación web sólo es una capa de maquillaje, una capa de presentación; esta capa esconde un servidor de correo que usará SMTP.

Hay otros servicios como el Telnet que nos permite acceder a ordenadores remotos y trabajar como si fueran nuestra propia máquina. Otro es el servicio de chat que funciona sobre el protocolo IRC y nos permite hablar en línea con gente de todo el mundo.

3.1.5. Aplicaciones web

El popular World Wide Web no sólo sirve para colgar páginas informativas sobre cualquier tema, con un diseño llamativo y atractivo.

Los ingenieros de software han descubierto en los navegadores el cliente perfecto; tiene un coste cero, la aplicación no necesita instalación en la estación de trabajo, tiene un mantenimiento casi nulo y para actualizar las versiones de la aplicación es suficiente con enviar un correo electrónico con la nueva URL[5]. Éstos son algunos de los motivos por los que la nueva tendencia en la ingeniería del software apunta a las aplicaciones web.

Navegadores

Programas para visualizar HTML, que es el lenguaje con el que se realizan las páginas web.

¿Qué es una aplicación web? Para entenderlo mejor, primero hay que tener claro qué es una aplicación cliente-servidor.

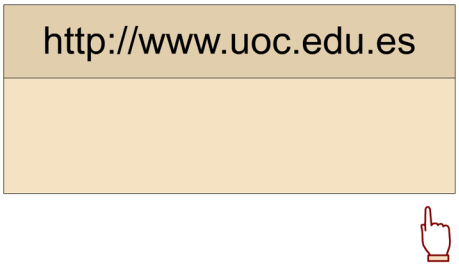
(5)URL: *uniform resource locators*. Es lo que se conoce como una dirección web. http://www.nombre.apellido

Todos conocemos el concepto clásico de aplicación, Excel, Word, Photoshop, Autocad, etc. Todo son aplicaciones. En ámbitos corporativos, donde muchos usuarios acceden a una misma aplicación, es donde tiene sentido utilizar aplicaciones cliente-servidor.

Una aplicación cliente-servidor consta de dos partes. Una parte denominada *servidora*, donde reside realmente la aplicación y los datos que manipulan todos los usuarios que tienen acceso a ella. Y otra parte denominada *cliente*, que únicamente sirve de presentación de la aplicación a la estación de trabajo y nos da la herramienta para interactuar con la aplicación. En el cliente se pueden hacer cálculos y procesar datos, eso depende del diseño que haya hecho el equipo de ingenieros responsables del desarrollo de la aplicación. Es decir, el procesamiento de los datos se divide entre la parte servidora y la parte cliente, pero a menudo el peso importante del procesamiento de los datos se hace en la parte servidora. Los datos, en el 99% de los casos, quedan guardados en el servidor o en servidores de bases de datos.

Una vez clara la idea de cliente-servidor podemos intuir que el World Wide Web nos da al cliente universal perfecto. Si los ingenieros de software son capaces de crear una aplicación que resida en un servidor de aplicaciones conectada a un servidor web, y desarrollar páginas web capaces de interactuar con esta aplicación servidora, habrán conseguido una aplicación web. Cualquier persona con un ordenador y una conexión a Internet podrá acceder a nuestra aplicación simplemente tecleando la URL adecuada en la barra de direcciones del navegador.

Esquema de una URL:



Hay dos grandes grupos de dominios, los territoriales y los genéricos, y dentro de los territoriales, patrocinados y no patrocinados:

- **Genéricos**: nos indican la temática o actividad de la web que hospedan. El más extendido es el .com o .co para actividades comerciales.

- **Territoriales**: tienen dos letras e identifican el país donde se hospeda la página. Ejemplos (.se, .uk, .it, .tk, .nl, etc.):
  - **Patrocinados**: tienen el apoyo económico de entidades patrocinadoras. Ejemplos de dominios genéricos patrocinados son: .cat, .eu, .aero, .jobs, .travel, etc.

  - **No patrocinados**: son no patrocinados los dominios: .name. info, .edu, .gov, .int.

Los dominios no patrocinados son gestionados por organismos internacionales como el ICANN, mientras que los patrocinados son gestionados también por el ICANN, pero con intervención de los organismos patrocinadores.

Algunos de los dominios más usados son:

Edu	Educación
Org	Organización no gubernamental
Gov	Organización, departamento, pertenecen a un gobierno oficial
Com/co	Indica actividad comercial, a pesar de que es el nombre genérico de toda página web
Biz	Indica actividad comercial (*bussiness*)
Neto	Proveedores de servicios de redes
Info	No tiene una orientación clara, se pueden encontrar .info para dar a conocer ideas, información personal, de empresas, etc.
Name	Está dedicado a usuarios particulares; no se admiten empresas ni organizaciones
Int	Para organizaciones con tratados internacionales u organizaciones no gubernamentales con estatus de observadores de la ONU

¡El navegador estará haciendo de cliente y, además, será cliente de cualquier aplicación! Sólo hay que cambiar la URL. No habrá que gastar ni un céntimo en licencias de cliente. No hay que instalar la aplicación al ordenador cliente, si sale una versión nueva se actualiza en el servidor web. O bien la nueva versión se pone en otra parte del servidor y se le da una nueva URL, de esta manera nos podemos conectar a las dos versiones, en función del URL que se teclee en el navegador.

La complejidad y el diseño de una aplicación web, aunque su capa gráfica sean páginas HTML, no tiene nada que ver con una típica página web, que no hace nada más que presentar información sin ningún tipo de dinamismo interactivo ni lógica de negocio detrás. En una aplicación web, detrás de la capa gráfica mostrada al cliente puede haber un complejo entramado de lógica de negocio.

Lógica de negocio

Se denomina *lógica de negocio* a todo el código que implementa los procesos de la aplicación. La lógica de negocio es la responsable que la aplicación haga lo que debe hacer.

Partes de una aplicación web

Depende del diseño del equipo de ingenieros de software, pero normalmente se aconseja dividir el diseño en presentación, datos y lógica de negocio. Es el llamado *modelo de tres capas*.

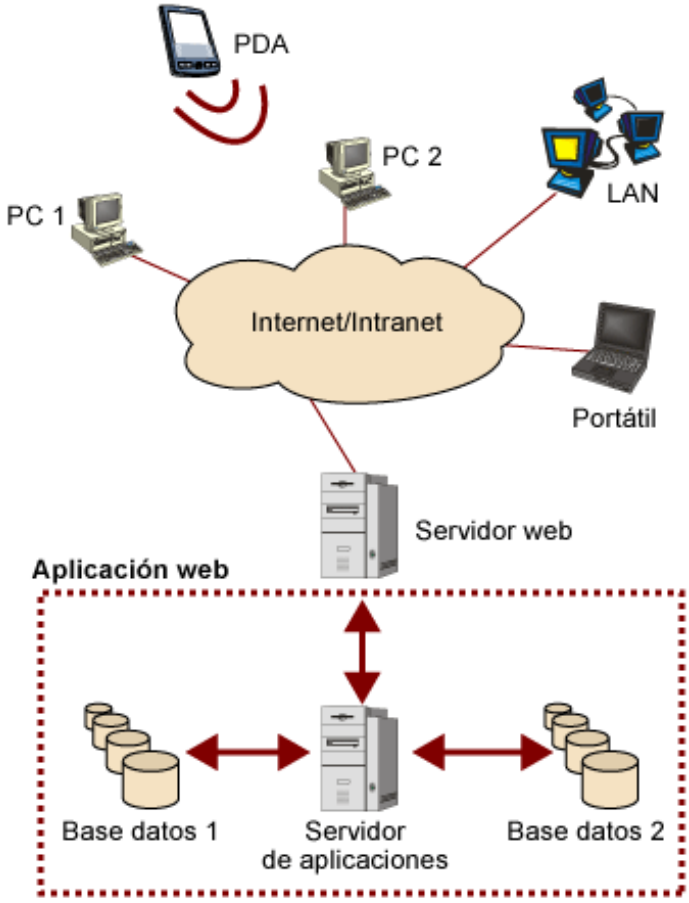
- **Capa de presentación**: son las páginas HTML a las cuales accede el cliente. Son un conjunto de páginas HTML (*hipertext markup language*), páginas JSP (*Java server pages*, una JSP es una página HTML con código Java hospedado), páginas ASP (*active server page*, una página ASP es una página HTML con Visual Basic hospedado o con ficheros con código C# asociados donde reside toda o parte de la lógica de negocio de la página), Scripts en JavaScript o Visual Basic (estas secuencias, aunque residen en el servidor, suelen ser la lógica de negocio que se ejecuta en el cliente).

- **Lógica de negocio**: sería el motor de la aplicación, lo que hace que funcione, que haga lo que se espera. Dentro de la lógica de negocio de las aplicaciones web hoy día, existen dos grandes tendencias: utilizar tecnología .NET (de Microsoft, para desarrollar en .NET es necesario pagar licencias) o bien utilizar tecnología J2EE (Java 2 Enterprise Edition de Sun microsystems; desarrollar con J2EE es gratis). Los lenguajes naturales para la lógica de negocio de las aplicaciones web son, para el caso de .NET, VisualBasic.NET o bien C#, y para el caso de J2EE, Java (Java estaría relacionada con las páginas JSP, y los otros dos con las páginas ASP o ASPX). Para el caso de aplicaciones realizadas con tecnología JAVA es habitual que parte de la lógica de negocio resida en *servlets* (un *servlet* es un fichero de código JAVA con código HTML hospedado, el caso recíproco de las JSP) y otros objetos más complejos como los JavaBeans, EJB (Enterprise Java Beans), etc.

- **Datos**: toda aplicación web seria tendrá detrás una base de datos y la lógica de negocios de la aplicación asociada, *triggers*, archivos SQL (*structured query language*), etc.

Al margen podemos ver el esquema de una aplicación web.

Modelo de la arquitectura de una aplicación web



Servidor web y bases de datos

Las estaciones de trabajo, los ordenadores portátiles y otros elementos como PDA pueden acceder a la aplicación web por medio de Internet o de la intranet. El acceso a la aplicación se realiza por el servidor web al cual se conectan los servidores de aplicaciones. Al mismo tiempo, el servidor de aplicaciones accede a las bases de datos a las que tenga acceso la aplicación web.

En el servidor de aplicaciones es donde reside realmente la aplicación web, es donde está la lógica de negocio: código C#, código Java, Servlets, JavaBeans, etc.

En el servidor web se encuentran las páginas que se enviarán al cliente para visualizar la interfaz de usuario. Páginas JSP, ASP, ASPx, ficheros JavaScript, etc.

En las bases de datos puede haber algún tipo de lógica de negocio si la base de datos es bastante potente para soportarla. Este tipo de ficheros de lógica de negocio se denominan *funciones* o *procedimientos almacenados* y contienen lógica de negocio de pura manipulación de datos.

Resumiendo, una aplicación web nos da las ventajas y los inconvenientes siguientes:

Ventajas	Inconvenientes
• No necesita instalación al cliente. • Familiaridad entre el usuario y el entorno. Se ejecuta en un navegador Internet al que el usuario ya está acostumbrado. • La potencia del lenguaje JAVA, VisualBasic.NET o C#. • Facilidad para atender clientes de diferente naturaleza, ordenadores, PDA, etc. • La facilidad en la reutilización de código, hay mucho código fuente abierto (sobre todo en J2EE) y otros recursos como HTML, JavaScript, etc. • Facilidad en la gestión de la concurrencia de usuarios.	• La bajada de páginas web agudiza la lentitud de las aplicaciones. • Arquitectura muy compleja. • Dependencia del navegador y de sus versiones.

3.1.6. Servicios web

Últimamente empieza a surgir con fuerza, impulsado por Microsoft, la idea de servicios web.

Un servicio web es una aplicación web con un diseño y una arquitectura particulares que permiten que cualquier usuario acceda a él y lo pueda utilizar como si fuera una parte de su aplicación web. Los servicios web pueden ser gratuitos o de pago.

La idea general es no reinventar la rueda, no reprogramar aquello que alguien ya ha programado anteriormente y que se ha comprobado que funciona correctamente. Es decir, el servicio web es un nuevo concepto de reutilización de código.

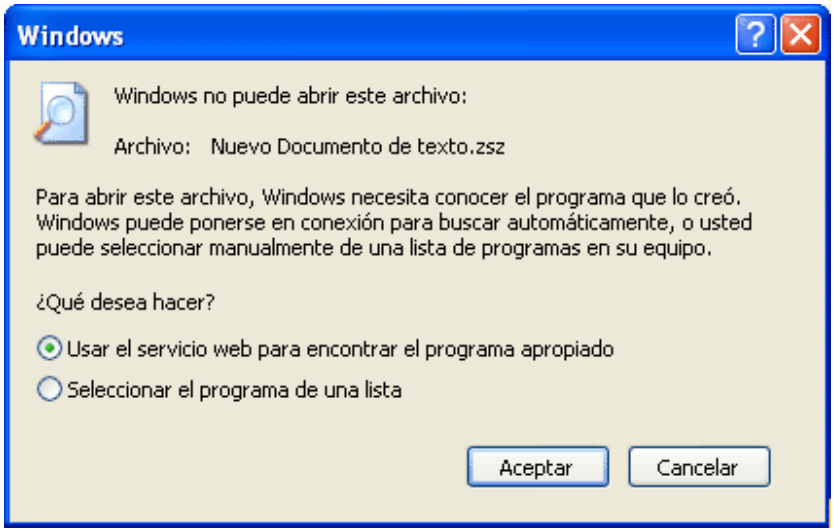
El componente reside en un servidor web y cualquier aplicación se puede comunicar con él para utilizarlo. Sólo hay que conocer las reglas de la comunicación con el servicio, qué debemos enviarle, con qué formato y qué nos devolverá el servicio.

Windows XP

Un ejemplo de servicio web lo tenemos en el mismo Windows XP. Si se tiene un fichero con un nombre de extensión desconocido y hacemos doble clic sobre el fichero, Windows

XP nos envía a un servicio web que nos muestra los programas que son compatibles con aquel tipo de fichero, nos dice dónde los podemos descargar, comprar, etc.

Como ejemplo de funcionamiento se puede crear un fichero de texto nuevo con el bloc de notas (inicio/ejecutar/notepad). Escribimos cualquier cosa, se graba y a continuación se cambia la extensión de un fichero de .txt a .zsz. Se hace doble clic en el fichero y Windows XP nos pregunta si queremos intentar encontrar, mediante un servicio web, un programa apropiado para abrir el fichero.



Windows XP nos pregunta si queremos acceder al servicio para encontrar una aplicación contable con el fichero de tipo ZSZweb.

Si aceptamos, se accede al servicio web y se obtiene como respuesta "*file unknown* (fichero desconocido)", ya que la extensión .zsz es inventada y no hay ninguna aplicación que la utilice.

3.2. Intranet y extranet

Ambas comparten la arquitectura con Internet y la única diferencia importante entre ambos conceptos es con respecto a la restricción del acceso.

3.2.1. Intranet

Una intranet es una red con una estructura basada en una filosofía arquitectónica como la de Internet, con la particularidad de que el acceso está restringido a un grupo específico de usuarios. En cambio, en Internet, todo el mundo es bienvenido.

La finalidad de la restricción del acceso a la intranet es la de garantizar la máxima seguridad posible para el intercambio de datos dentro de una institución u organización corporativa. Esta privacidad se consigue mediante elementos propios de las redes como los *firewalls* o cortafuegos.

Hemos dicho que denominábamos *pasarela* (*gateway*) al nodo, al elemento físico (hardware), que unía nuestra red con el mundo exterior. Desde el punto de vista del software, esta unión la realiza un tipo de software que denominamos *firewall* o cortafuegos. Cualquier mensaje que entre o salga de la LAN deberá superar la supervisión del cortafuegos. Los cortafuegos pueden restringir

las comunicaciones en función de su origen, de su naturaleza (pueden cortar un tipo de servicios determinado: FTP, IRC, etc.). También pueden albergar potentes antivirus para evitar infecciones masivas dentro de la LAN.

3.2.2. Extranet

Una extranet es una red privada con una estructura basada en Internet, generalmente corporativa, a la cual se permite la entrada, con identificación previa, de algunas personas ajenas al grupo propietario de la extranet. Normalmente, acceden a ella clientes, proveedores, socios, etc.

La única distinción entre intranet y extranet es que en la intranet el uso es exclusivo para los usuarios que la componen. En cambio, en una extranet se permite el acceso de usuarios externos, sea porque tienen visibilidad de la intranet entera o sólo de una parte.

Empresa de logística

Pongamos como ejemplo de esto una empresa de logística que gestione el envío de paquetes de sus clientes. Tiene una red corporativa a la cual deja entrar a sus clientes para poder contratar sus servicios y realizar el seguimiento de los paquetes que están en ruta y poder saber en todo momento dónde se encuentran o si ya se han entregado. También, y a efectos comerciales, dispone de una página web donde da información de sus servicios.

Ante esta situación, vemos que existe la necesidad de distinguir entre tres zonas en la red corporativa: una privada a la cual sólo tendrán acceso los usuarios que forman parte de la propia intranet para entrar/actualizar datos de envío, asignar rutas, etc., una segunda a la que se accederá con la identificación previa de los clientes para que puedan consultar el estado de sus pedidos, facturas, etc. y una tercera parte libre, sin restricciones, donde todo el mundo puede entrar y ver los servicios que la empresa ofrece.

- **Zona privada**: estaría formada por las estaciones de trabajo de la empresa, las impresoras de red, los servidores de aplicaciones, etc.
- **Zona de clientes**: sería una aplicación web que residiría en algún servidor de aplicaciones conectado a algún servidor web.
- **Zona pública o desmilitarizada**: serían un conjunto de páginas HTML que podrían residir en un servidor web.

4. Dimensionado de redes

Llegados a este punto del módulo, ya tenéis amplios conocimientos de redes. La unidad está dedicada a detallar aquellos puntos más conflictivos a la hora de hacer el diseño de una red; aunque este diseño debe recaer en personal técnico especializado, es necesario que el profesional de los sistemas de información o documentación domine el argot o conozca conceptualmente qué tiene en las manos, qué implica un buen diseño de red y con qué dificultades y limitaciones se puede encontrar.

A lo largo de la unidad se habla de manera extensa del concepto de ancho de banda porque en definitiva es el que en buena medida es el responsable de la satisfacción de los usuarios de la red. Finalmente, se dan algunas URL donde se puede encontrar información para la contratación de líneas o servicios de acceso a Internet.

4.1. Dimensionado

Para que la red sea un medio útil con el que trabajar, es necesario que el diseño sea acertado en todos los parámetros.

A la hora de diseñar una red, hay una gran cantidad de parámetros y conceptos que hay que tener en cuenta y que repercutirán en su buen funcionamiento.

Se ha de dejar el diseño en manos de técnicos especializados, a los que es fundamental dar toda la información de las necesidades de la entidad que utilizará la red, todas sus particularidades, software que se utiliza, número de usuarios, etc. Sólo de esta manera se podrá afinar al máximo en el diseño.

En el diseño hay que tener en cuenta el número de terminales que accederán a la red, qué aplicaciones accederán, a qué servicios externos accederemos, hacer cálculos aproximados del número de usuarios concurrentes que pueden utilizar la red, etc. Con todos estos datos, la persona o el grupo responsable decidirá la topología, el hardware y el software más adecuado para que funcione la red.

Como ya se ha apuntado, hay un parámetro clave en el dimensionado: el ancho de banda.

4.1.1. Ancho de banda

El ancho de banda es un parámetro clave para cualquier sistema de telecomunicación, ya que nos dice la capacidad de transmitir información que tiene un sistema.

Un ancho de banda insuficiente provocará lentitud y mucha insatisfacción a los usuarios, por más que las aplicaciones sean excelentes. Un ancho de banda excesivo supone un gasto injustificado, a golpes de cantidades muy elevadas.

Para tener una idea conceptual de qué significa ancho de banda, a continuación se explica para un sistema analógico y después se extrapolará a un sistema digital.

Sistemas analógicos

En los sistemas analógicos el ancho de banda se mide en hercios (Hz), ya que el ancho de los canales de transmisión ocupan más o menos hercios dependiendo de su naturaleza espectral.

Una señal ocupa un determinado rango de frecuencias; la distancia que hay entre la frecuencia más alta y la más baja es lo que se denomina *ancho de banda de la señal*. Para poder transmitirla correctamente, es necesario que el sistema soporte un ancho de banda igual o superior a la de la señal.

Analizador de espectros. En la pantalla, podemos ver el espectro de una señal.

La voz humana tiene un rango espectral que va desde los 30 hasta los 4.000 Hz, aproximadamente, y por ello un canal telefónico tiene un ancho de banda que va de los 300 a los 3.400 Hz, es decir, tiene un ancho de banda de 3.400 – 300 = 3.100 Hz o 3,1 kHz. Como se puede ver, para un canal de telefonía se pierden algunas frecuencias altas (3.400 a 4.000 Hz) y algunas bajas (30 a 300 Hz); esto provoca que la voz por teléfono no suene exactamente igual que cuando la recibimos por el aire, pero bastante clara para entender el mensaje.

Ejemplo de ancho de banda

Otro ejemplo de ancho de banda lo encontramos en los equipos de alta fidelidad: se considera alta fidelidad si el equipo es capaz de responder sin distorsión a frecuencias de 20 Hz a 20.000 Hz (20 kHz). Se puede comprobar en las especificaciones de los equipos de alta fidelidad (HI-FI, *high fidelity*) que tanto el amplificador, como los altavoces deben soportar este rango de frecuencias.

Sistemas digitales

Son los que realmente nos interesan, ya que por las redes viajan unos y ceros (bits). Aunque físicamente viajan ondas analógicas, en la naturaleza no existen las ondas digitales, la modulación de estas ondas transportan bits (modulación digital[6]).

[6]Durante todo este apartado se ha evitado entrar en el detalle de las modulaciones digitales, ya que, para entenderlas, hace falta una base matemática fuerte. El concepto, sin embargo, es el mismo que en la modulación AM, ya vista.

Por lo tanto, ya no tiene sentido hablar de la capacidad de los sistemas digitales como la capacidad de albergar hercios en su interior, sino que ahora lo que queremos saber es cuántos bits es capaz de transmitir. El ancho de banda digital se mide en bits por segundo (bps).

Cálculo del ancho de banda

Disponer del ancho de bando necesario para que todos los usuarios de la red estén satisfechos es imposible. El ancho de banda es un bien caro y escaso, y por mucho que se ajusten los cálculos habrá usuarios que lo encontrarán insuficiente.

Es necesario realizar un cálculo razonable y, sobre todo, evitar los cuellos de botella que se puedan producir por culpa de la infraestructura del hardware. Una vez se tenga un diseño y la implantación de la red sea razonable, no se podrá evitar que en un momento concreto, inconscientemente, un número importante de los usuarios de la Red tengan la necesidad irrefrenable de bajar de Internet el último CD de su cantante preferido, adquirir la última versión de algún software, etc. Y estas situaciones, evidentemente, moderan la Red.

Hay métodos para controlar la cantidad de información que circula por la Red: se puede restringir el acceso a Internet, se puede limitar el acceso al servicio FTP para evitar bajadas de ficheros muy grandes que ocupan mucho ancho de banda, también se pueden limitar las dimensiones de los ficheros adjuntados al correo electrónico, etc. Pero aunque se "controle" el volumen de información y que los usuarios hagan un buen uso de la Red, siempre existe la posibilidad de que en un momento determinado se sature su capacidad y el rendimiento caiga.

Sabiendo que el sistema perfecto no se puede conseguir, hay que procurar que las limitaciones sean mínimas, se pueda utilizar la red sin que nadie desespere y que cuando esto suceda no sea culpa de un mal diseño.

Algunos de los elementos que hay que tener en cuenta para no provocar un cuello de botella son:

- **Medio**: el cable, sea fibra, coaxial o par trenzado, debe soportar velocidades de transmisión suficientes para el sistema. Los tramos de cable no han de ser nunca más largos de lo que las especificaciones del mismo cable recomiendan. Si se trata de un tramo WIFI, es necesario que el hardware de transmisión sea compatible con las normas de transmisión según la velocidad que se busque, por ejemplo, para 54 Mbps, es necesario que soporte la norma 802.11g.

- **Tarjetas de red**: en cada PC o portátil habrá tarjetas de acceso a la red y es necesario que éstas soporten velocidades de transmisión adecuadas.

- **Líneas, ancho de banda contratado**: si hay acceso a Internet o se tienen que unir diferentes sedes mediante la contratación de líneas punto a punto o dedicadas, hay que contratar un ancho de banda de transmisión suficiente y con bastantes garantías de servicio.

- **Hardware de red (*routers*, conmutadores, concentradores)**: todos los elementos de la Red tienen que soportar el mínimo de velocidad que se haya establecido por el equipo de diseño. Estos elementos son especialmente susceptibles de formar un cuello de botella, ya que reciben transmisiones de puntos diferentes de la Red y los tienen que redireccionar hacia su destino. Hace falta que la capacidad de tráfico de salida de estos elementos sea proporcional al máximo tráfico posible en sus entradas.

- **Software (cortafuegos, servidores web, servidores de aplicaciones, servidores de bases de datos, etc.)**: el software debe estar de acuerdo con el uso que se le da.
  - Los servidores de aplicaciones se han de configurar correctamente y ser capaces de atender de manera concurrente un mínimo del 10% del total de usuarios que tienen acceso a la aplicación.

  - Los servidores web, como en el caso anterior, deben ser capaces de atender de modo concurrente un número suficiente de usuarios.

  - El cortafuegos ha de estar configurado correctamente, de manera que dé seguridad a la red, pero que no castigue en exceso su eficiencia.

  - Servidores de bases de datos: cualquier gestor de bases de datos no puede atender a un gran número de usuarios concurrentes y gestionar simultáneamente bases de datos complejas y con gran capacidad. Hay que elegir muy bien las bases de datos que se tienen que utilizar y, sobre todo, el equipo de ingenieros de software responsable de las aplicaciones que acceden tienen que programar el acceso a los datos de manera eficiente. La presencia de un DBA (*data base administrator*) también ayudará a tener un buen mantenimiento de la base de datos, tenerla ordenada y que los accesos a ella sean más fluidos.

- **Hardware**: los servidores de aplicaciones, web y bases de datos han de estar en máquinas con bastante potencia de cálculo, bastante memoria, etc.

El conjunto de estos factores, junto con el buen funcionamiento del sistema operativo de red y una topología acertada, garantizarán que la red sea eficiente, eficaz, escalable y mantenible.

Aunque se haga un proyecto para diseñar la red con todas las garantías y, teniendo en cuenta todo lo que se ha dicho anteriormente, todavía existe un último parámetro que los limita todos, de hecho, siempre es el más restrictivo: el presupuesto. Desde el punto de vista técnico, no se tiene muy en cuenta, pero

a la hora de hacer cualquier implementación es el concepto clave. Respetar los puntos anteriores resulta caro, muy caro. Sólo se puede conseguir si, además de un buen equipo de diseño, hay bastante presupuesto, algo que nunca suele pasar; por ello, desgraciadamente el principal limitador en la calidad de la red será el presupuesto del que se disponga para implementarla y mantenerla. El sistema perfecto, sin cuellos de botella y bastante ancho de banda para todo el mundo, el sistema con riesgo cero tiene un coste infinito. Por lo tanto, habrá que llegar a una solución de compromiso entre la calidad de servicio y lo que se está dispuesto a pagar.

4.2. Servicios comerciales de transporte de datos

Algunas veces no es suficiente cablear un edificio, una sala o una casa. En otras, es necesario que la red contenga edificios situados en diferentes ciudades o edificios de una misma ciudad muy distantes.

En estos casos, se puede pedir permiso de obra pública, excavar y cablear el suelo o bien contratar líneas en empresas de telecomunicaciones. Es evidente que la segunda solución es la única factible.

Las empresas de telecomunicaciones ofrecen una gama de soluciones variada, desde soluciones estándar para casos poco conflictivos hasta carísimas soluciones particularizadas.

Si el caso lo requiere, se pueden unir diferentes edificios contratando diferentes tipos de servicios:

- **Líneas punto a punto dedicadas**: una línea con origen y destino en los dos edificios que se van a unir y de uso exclusivo para quienes la contrate.

- **Radioenlaces punto a punto**: la misma filosofía que la línea punto a punto, pero en lugar de cable se utilizan antenas para unir los edificios.

- **Ancho de banda**: se contrata un determinado ancho de banda entre dos destinos, sin tener la exclusividad de la línea. La compañía proveedora garantiza el ancho de banda mínimo, de manera que, normalmente, si la línea no va al máximo de su capacidad, se puede transmitir más del caudal contratado. Si se transmite más cantidad de información de la contratada, no se garantiza que llegue, ya que en el momento en el que la línea necesite el caudal "sobrante", retirará la información que viaja "gratis" y dejará sólo el caudal contratado.

- Si la red es pequeña y hay pocos usuarios, puede ser que haya suficiente con la contratación de una **línea ADSL**.

Servicios hay muchos y, como las condiciones, varían en poco tiempo. La única manera de saber qué se ofrece hoy día es pedir información a las operadoras de servicios de telecomunicación.

Operadoras de servicios de telecomunicación

- Telefónica de España S. A.
- Abertis.
- Ono.
- British Telecom.

Cuando se quiera acceso a Internet, también habrá que recurrir a empresas proveedoras, denominadas ISP (*Internet service provider*).

Contratar líneas

A la hora de contratar el acceso a Internet, dependerá básicamente del número de usuarios que tendrán acceso a la Red. Si son pocos, con una línea ADSL habrá suficiente; si el número es grande, habrá que contratar líneas con más ancho de banda, ya que se comparte. Si se contratan 20 Mb y hay cien usuarios, suponiendo que exista una concurrencia del 10% (concurrencia **muy** baja tratándose de Internet), nos quedan como mucho 2 Mb, aproximadamente. La mitad del mínimo ofertado por las compañías de servicios de telecomunicaciones como ADSL doméstica. Claramente insuficiente para uso profesional.

Los servicios que ofrecen los ISP son muchos, variados y cambian constantemente, tanto en prestaciones como en precios. Como en los servicios de telecomunicaciones, la única manera de saber el estado del arte es pidiendo información.

Proveedores de servicios de Internet

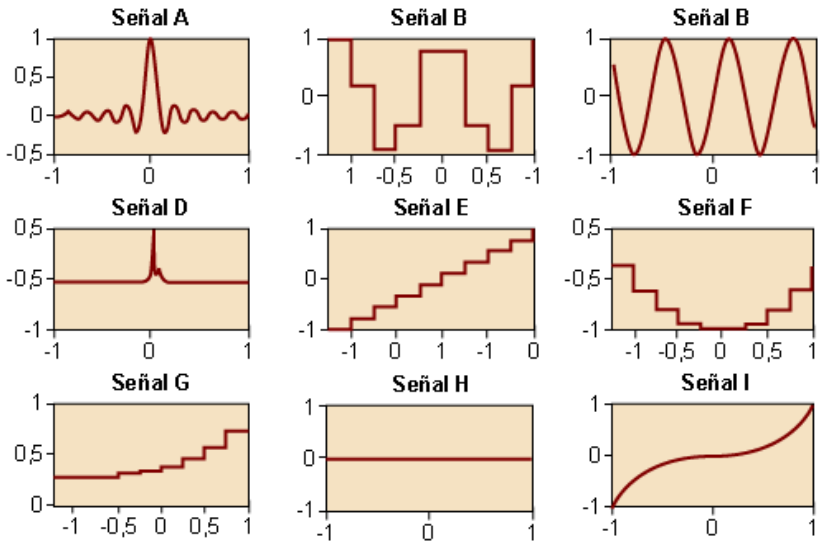
- Telefónica de España S. A.
- Ono.
- France Telecom.
- Jazztel.
- Ya.
- British Telecom.

Ejercicios de autoevaluación

1. Señales digitales/analógicas

De las señales del siguiente gráfico, ¿cuáles son analógicas y cuáles digitales?

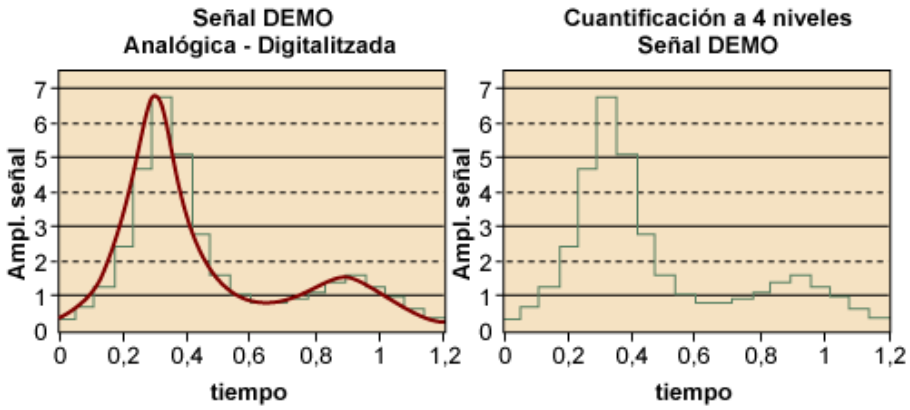
De las señales analógicas, hay una de la que en el proceso de muestreo sólo habría que tomar una única muestra, ¿cuál es? ¿Por qué?



2. Cuantificación, codificación, ruido

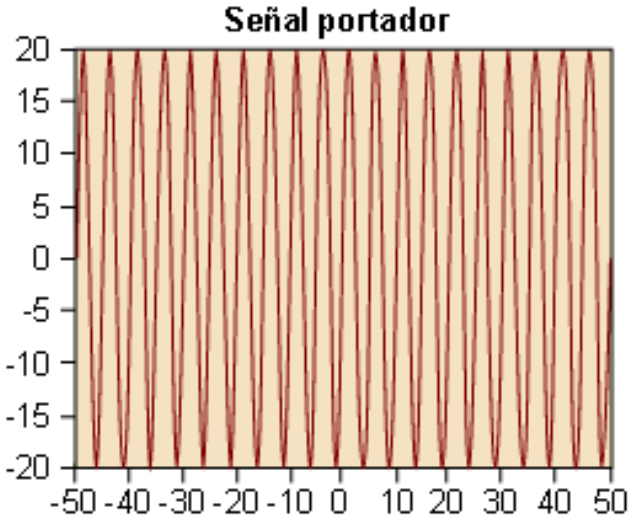
En el gráfico inferior está la señal DEMO ya muestreada según el criterio de Nyquist.

a) Cuantificad a cuatro niveles la señal digitalizada resultada del muestreo (tomad como referencia la gráfica de la derecha).
b) Escribid el tren de bits resultante de la señal digitalizada y cuantificada.
c) ¿Al cuantificar a cuatro niveles, la señal resultante es más o menos fiel a la señal DEMO original que la señal cuantificada a ocho niveles vista en los apuntes de la asignatura?
d) ¿Al cuantificar a cuatro niveles, la señal resultante es más o menos resistente al ruido que la señal cuantificada a ocho niveles que hemos visto en los apuntes de la asignatura?
e) ¿Hasta qué nivel de ruido la señal que se ha obtenido se mantiene inmune?

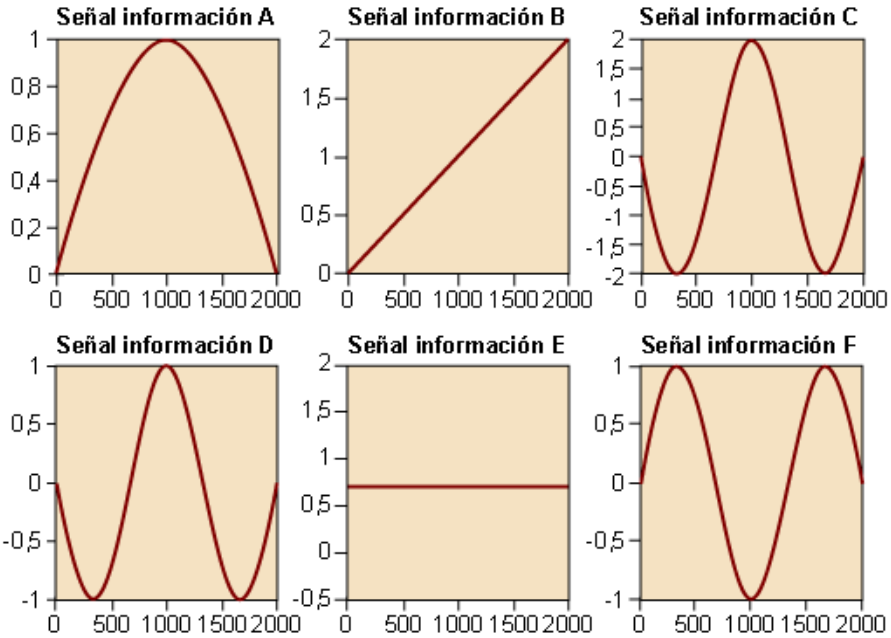


3. Modulación

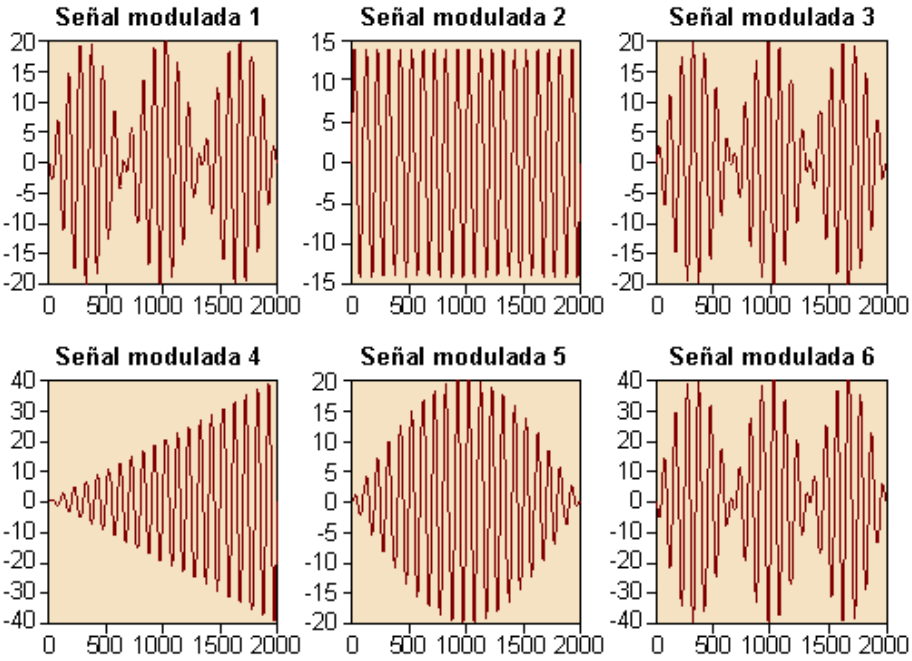
Se dispone de la señal portadora siguiente:



y de las señales de información siguientes:



Cada una de las señales A, B, C, D, E y F modularán la señal portadora para dar lugar a las señales moduladas 1, 2, 3, 4, 5 y 6.



a) ¿A qué señal de información corresponde cada señal modulada?
b) ¿Qué tipo de modulación se ha utilizado?

4. Tipos de red

Las redes se pueden clasificar de muchas maneras diferentes; en el módulo se ha aprendido a clasificarlas tipológicamente en función de la extensión. A continuación se presenta una serie de situaciones, ¿a qué tipo de red corresponde cada una?

1. Se quiere diseñar una red para la comunicación de las diferentes facultades de una universidad. Todas las facultades están en un mismo campus universitario.
2. Vista la necesidad de compartir información y recursos (proyectos, documentación, programas de CAD, trazadores o *plotters*, impresoras de gran formato, etc.) entre los diferentes miembros de un estudio de arquitectos, se decide implementar una red para comunicar todos sus puestos de trabajo.
3. Una entidad bancaria decide completar su vieja red X.25 con una red con tecnología TCP/IP para dotar de correo electrónico a todos sus empleados; la red también se utilizará para acceder a la intranet de la empresa. La entidad bancaria tiene una gran capilaridad de oficinas distribuidas por todo el territorio nacional.
4. Se decide implementar un cableado en forma de bus para un edificio de cuatro plantas que ha de dar servicio al personal administrativo de un ayuntamiento.
5. Una empresa de fabricación de semielaborados para la industria del plástico decide actualizar su red informática. La empresa tiene un edificio de oficinas situado a pocos metros de la fábrica. Es necesario que el cableado una tanto oficinas, como algunos puestos de trabajo de la fábrica para poder llevar un control en línea de almacenamientos y la gestión de las materias primas que aportan los proveedores.

5. Colisiones y eficiencia de una red

Una red ethernet tiene un protocolo de acceso basado en CSMA/CD.

1. ¿Es posible que haya colisiones en un protocolo de acceso CSMA/CD? ¿Por qué?
2. ¿Cuál de estas situaciones haría bajar la eficiencia de la red y por qué?
   a) Pocos usuarios que utilizan poco la red pero transmiten gran cantidad de información cada vez.
   b) Muchos usuarios que transmiten mensajes breves pero que acceden a la red con mucha frecuencia.
   c) La presencia de impresoras conectadas a la red.
   d) Un software cortafuegos en la pasarela.

6. Diseño de red doméstica

Con la ayuda de una hipoteca y los ahorros de toda una vida, el señor y la señora X acaban de comprar una bonita casa con jardín y piscina donde han decidido trasladarse con toda la familia.

La casa tiene cinco habitaciones, comedor, cocina, tres lavabos, estudio, garaje, bodega, dos terrazas, jardín, piscina, etc. hay tomas para la conexión telefónica en todas las habitaciones, cocina, comedor y estudio.

Tanto el señor, como la señora X y dos de sus tres hijos necesitan acceso a Internet. Cada hijo tiene un ordenador en su habitación, el señor y la señora X compartirán un ordenador situado en una de las habitaciones que habilitarán como despacho. Igualmente, tienen una impresora multifunción con conexión en red que estará en el despacho. Los negocios del señor X no lo dejan vivir sin conectarse permanentemente a la información financiera que extrae de Internet, por ello quiere poder conectarse en Internet desde el portátil cuando está en el jardín, en la piscina, sentado en el sofá del comedor mirando el televisor o cuando sube al estudio para aislarse y escuchar música.

Ni el señor ni la señora X quieren hacer agujeros en las paredes, ni están dispuestos a romper la estética de la pared clavando clavos para sujetar cables.

1. ¿Qué tipo de red necesitan?
2. ¿Qué elementos de hardware han de comprar para poder satisfacer todas sus pretensiones?
3. ¿Cuál es el lugar ideal para colocar pasarelas?
4. Sabiendo que la planta baja tiene 110 m^2, la primera planta 90 m^2, el estudio 35 m^2, y el total de la parcela 520 m^2, ¿habrá que poner alguna antena repetidora?
5. El hijo pequeño se aficiona rápidamente a los juegos de consola y el comedor de su casa le queda pequeño, se quiere abrir al mundo y demostrarle su habilidad. ¿Qué habría que añadir para poder conectar la consola a Internet?

7. Relación de conceptos internet-intranet-extranet

Relacionad los conceptos siguientes, las relaciones no son 1 a 1, a cada concepto de la columna izquierda pueden corresponder 1 o más conceptos en la columna de la derecha:

	SMTP
	FTP
Internet	IP
World Wide Web	POP
Correo electrónico	TCP/IP
TCP/IP	TCP
Chat	IRC
Transferencia de ficheros	HTTP
	HTTPS
	UDP

8. Aplicaciones web

La World Wide Web, aparte de ser una fuente inagotable de información, tiende cada vez más a ser una herramienta para agilizar procesos, solicitudes, etc. que hasta la aparición de las aplicaciones web casi no existían.

Buscad, enumerad y describid muy brevemente al menos cinco aplicaciones web que estén en funcionamiento ahora mismo en la World Wide Web, en la intranet de vuestra empresa o en cualquier otra organización.

9. Ancho de banda

1. Buscad el ancho de banda del contrato con vuestro proveedor de Internet.

2. Buscad una página en Internet que os compruebe vuestro ancho de banda real y comparad los resultados con el ancho de bando contratado.

¿Coincide el valor real que nos ha dado el test con el teóricamente contratado? ¿Por qué?

Solucionario

Ejercicios de autoevaluación

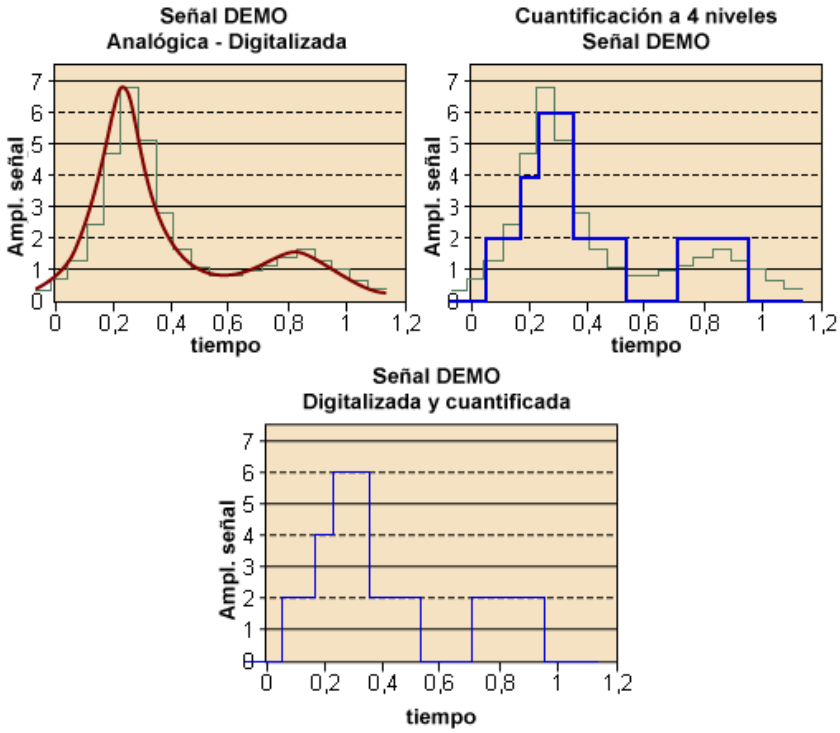
1.

Señal	Digital/analógico
Señal A	Analógico
Señal B	Digital
Señal C	Analógico
Señal D	Analógico
Señal E	Digital
Señal F	Digital
Señal G	Digital
Señal H	Analógico
Señal I	Analógico

La señal H es analógica y se podría muestrear tomando una única muestra, ya que es constante, tiene frecuencia 0 y, por lo tanto, con una única muestra ya se cumple el criterio de Nyquist. La muestra nos da la amplitud de la señal constante, que es el único dato que nos hace falta para reproducirlo exactamente.

2.

a) En la gráfica superior derecha podemos ver el proceso de cuantificación a cuatro niveles, en la gráfica de abajo el resultado final.



b) 00000101101111010101000000010101010000000

c) Menos fiel. Cuantos menos niveles de cuantificación haya, más alejada estará la señal digital de la señal analógica.

d) Más resistente al ruido. Al haber menos niveles de cuantificación, hay más distancia entre los niveles y es necesario más ruido para hacer que la señal pase de un nivel a otro.

e) La señal será inmune a cualquier ruido que no tenga amplitudes superiores a 1 o inferiores a –1.

3.

a)

Señal modulada	Señal de información
Señal 1	Señal D
Señal 2	Señal E
Señal 3	Señal F
Señal 4	Señal B
Señal 5	Señal A
Señal 6	Señal C

b)

Las modulaciones son de amplitud: AM.

4.

1. MAN.
2. LAN.
3. WAN.
4. LAN.
5. LAN.

5.

1. Sí. A causa del tiempo que tarda el mensaje en llegar a los nodos de la red desde que sale del emisor, si uno de los nodos quiere transmitir, se puede dar el caso de que en el momento de la escucha al nodo no le haya llegado el mensaje y considere el bus como libre.

2.a Hace bajar la eficiencia de la red, ya que la situación genera un aumento del tráfico en la red, al mismo tiempo que aumenta la probabilidad de colisiones.

2. b Hace bajar la eficiencia de la red, ya que la situación genera un aumento del tráfico en la red, al mismo tiempo que aumenta la probabilidad de colisiones.

2.c No afecta al tráfico, por lo tanto, no aumenta la probabilidad de colisiones y la eficiencia no se ve afectada.

2.d No afecta al tráfico, por lo tanto, no aumenta la probabilidad de colisiones y la eficiencia no se ve afectada.

6.

1. Una red sin hilos o WIFI.

2. La solución no es única. Basándonos en la ilustración 20, una posible solución sería:

- Un direccionador con características WIFI.
- Tarjeta de red.
- 2 tarjetas de red WIFI.
- Una tarjeta de red WIFI PCMIA.
- Cable para unir el direccionador y la impresora.
- Cable para unir el direccionador y el ordenador de la habitación donde está el direccionador.

3. El direccionador hará funciones de pasarela. La red WIFI es delicada y el punto crítico es la atenuación por la distancia, por lo tanto, será positivo poner el direccionador en la habitación más céntrica de la casa, con el fin de minimizar las distancias en los otros ordenadores.

4. Depende de la potencia de emisión del direccionador y de las tarjetas que se compren, en las especificaciones de los productos podemos encontrar información. En las webs de los fabricantes acostumbra a haber este tipo de datos.

5. Sólo habrá que conectar un punto de acceso a la red WIFI en la consola.

7.

Internet	TCP/IP
World Wide Web	HTTP HTTPS
Correo electrónico	SMTP PULPO
TCP/IP	IP TCP UDP
Chat	IRC
Transferencia de ficheros	FTP

Reflexión

NOTA: Para que todo funcione, hay que contratar acceso a Internet, por ejemplo, una línea ADSL. También habrá que configurar la red, dar IP a cada ordenador, etc.

8.

1. El campus de la UOC.
2. La posibilidad de reservar habitaciones de hotel, alquilar coches, etc.
3. Los servicios de gestión de las cuentas en línea que ofrecen la mayoría de bancos y cajas de ahorros.
4. La posibilidad de comprar billetes de avión que ofrecen muchas páginas web.
5. Las aplicaciones web que hay en la intranet de muchas empresas para el control y gestión de las vacaciones, horas trabajadas, etc.

9.

1. La solución depende de cada caso.
2. La prueba del test depende de cada caso. Se puede hacer el test desde numerosas páginas de Internet, a alguna de éstas podemos acceder desde el enlace siguiente: <http://www.adsl4ever.com/test.>

La velocidad contratada es siempre la velocidad máxima a la que podemos operar, la velocidad real siempre es inferior. Eso pasa porque aparte de la información también se transmiten los bits del protocolo TCP/IP (que puede llegar a ocupar un 20% de la anchura de banda contratada) y otra información para garantizar la comunicación con éxito. En general, los tests sólo miden la cantidad de información útil transmitida. En la reducción de la capacidad real también pueden influir otros aspectos más aleatorios como la congestión de las líneas, su calidad, interferencias, etc.

Bibliografía

Tanenbaum, Andrew S. (2003). *Redes de computadoras* (4.ª ed.). México [etc.]: Pearson Educación cop.

Inteligencia artificial

Miquel Barceló García

PID_00153113

Universitat Oberta de Catalunya
www.uoc.edu

Índice

Introducción

La inteligencia artificial es, con toda seguridad, el intento más ambicioso de los que aborda la informática. Superado el estadio de construir máquinas más fuertes y rápidas que su propio constructor, al ser humano todavía le queda el reto de construir máquinas que al menos parezcan inteligentes, como se supone que lo somos nosotros.

Intentos de la inteligencia artificial

Después de intentos un poco esotéricos como los *homúnculos* de Paracels o el *golem* que obedece al rabino de Praga por el efecto de una palabra mágica, la inteligencia artificial intenta traer a la realidad tecnológica la construcción de máquinas y robots que se comporten inteligentemente.

Aunque pueda parecer un campo emparentado con la ciencia-ficción, la realidad es que hoy día disponemos de programas que superan a los humanos en algunos aspectos de lo que consideramos actividad en cierta manera inteligente.

La inteligencia es difícil de definir, y por ello nadie pretende construir programas inteligentes sino, simplemente, programas con un comportamiento que calificaríamos de inteligente en un ser humano. Es un poco parecido a conseguir moverse por las aguas marinas como hace, por ejemplo, un submarino, pero que pocos se atreverían a etiquetar de "nadar" lo que hace un submarino...

Este módulo presenta los **problemas centrales** de la inteligencia artificial y sus **técnicas particulares**, y explica con más detalle algunos casos concretos como los resultados obtenidos con los sistemas expertos y el renacimiento de la tecnología de las redes neuronales.

Objetivos

Los objetivos que el estudiante puede alcanzar son los siguientes:

1. Definir y delimitar el amplio campo de la IA, indicando sus principales aplicaciones.

2. Introducirse en el estudio de las principales técnicas que utiliza la IA.

3. Asumir la problemática de cómo se representa el conocimiento y las diferentes propuestas en curso.

4. Conocer con un cierto detalle las posibilidades de los sistemas expertos, las técnicas específicas y los ejemplos más característicos.

5. Comentar el renacimiento de la técnica de las redes neuronales y exponer las principales características.

6. Saber cuáles son los lenguajes específicos utilizados en la IA.

1. Definición, objetivos y otras denominaciones

Es bastante difícil definir exactamente el contenido y el alcance de la inteligencia artificial (IA), a pesar de los largos años transcurridos desde que John McCarthy acuñara el nombre en el año 1956 en la Darmouth Conference en Hannover (New Hampshire, Estados Unidos).

Quizá una de las caracterizaciones más breves y sencillas es aquella que, parafraseando Marvin Minsky, uno de los expertos e investigadores más famosos de la IA, asigna a la inteligencia artificial la "realización de sistemas informáticos con un comportamiento que en el ser humano calificaríamos de inteligente".

1.1. El test de Turing

A este efecto, es famoso el denominado **test de Turing**, establecido por Alan Turing en un artículo que data de 1950 ("Computing machinery and intelligence") para proponer la realización de un experimento que permita discernir el carácter inteligente o no del comportamiento de una máquina.

El test mencionado parte de un juego en el que un interrogador debe averiguar el sexo de dos interlocutores, A y B, situados en una habitación separada y que, aunque ambos dicen que son mujeres, en realidad son un hombre y una mujer. En la propuesta original de Turing se trata de sustituir a la mujer por un ordenador y esto se ha generalizado después en la forma siguiente: el interrogador ha de averiguar quién es la máquina a partir de la conversación con los dos interlocutores, una persona y un ordenador, aunque ambos dicen ser personas.

Este objetivo se ha de conseguir a pesar de saber que ambos interlocutores no están obligados a decir la verdad y que, por ejemplo, la máquina puede decidir dar un resultado erróneo a una multiplicación e, incluso, comunicarlo después de muchos segundos o minutos de haberlo obtenido para engañar al interrogador.

En la hipótesis optimista del mismo Turing, hacia el año 2000 se dispondría de ordenadores suficientemente potentes "para hacerles jugar tan bien el mencionado juego que un interrogador normal no tendrá más del 70% de posibilidades de efectuar la identificación correcta al cabo de cinco minutos de haber planteado las preguntas".

Si fuera así, estaríamos ante una máquina verdaderamente "inteligente" o, cuando menos, que se sabe hacer pasar por tal.

Es evidente que la predicción de Turing peca de un optimismo exagerado, algo que, como veremos, es un error muy frecuente en los inicios de la IA.

En realidad, el problema no afecta sólo a la potencia del ordenador, sino a las potencialidades de su programación para poder mantener una conducta inteligente.

1.2. Objetivos centrales de la investigación en inteligencia artificial

Actualmente se puede afirmar que cuando se investiga en el campo de la inteligencia artificial se pueden perseguir dos objetivos complementarios que ponen el énfasis respectivamente en el aspecto teórico o tecnológico de la IA.

Un primer objetivo es **el estudio de los procesos cognoscitivos en general**, lo cual justificaría la definición de Patrick J. Haye en *El estudio de la inteligencia como computación* (1973) que se orienta a la consideración de la inteligencia artificial como estudio de la conducta humana inteligente.

En este estudio es fundamental la ayuda obtenida de la informática como herramienta de gran potencia en la representación de cualquier sistema de símbolos como el que utiliza la actividad racional. De aquí los múltiples puntos de contacto de la inteligencia artificial con otras ciencias como la neurofisiología, la psicología, la lógica formal, la lingüística, etc.

Otro objetivo central de la IA es **el intento de obtener sistemas automáticos capaces de llevar a cabo tareas reservadas, incluso ahora, a los seres humanos**.

Con este enfoque, la IA aparece como una disciplina eminentemente tecnológica que persigue la construcción de máquinas y programas capaces de realizar tareas complejas con una habilidad y eficiencia iguales o superiores a las que consigue el ser humano.

1.3. Otras definiciones

Los dos objetivos se suceden en las distintas definiciones que los especialistas más importantes de la IA han ido dando a los contenidos de esta ciencia.

Para Marvin Minsky, profesor del MIT (Massachussetts Institute of Technology) y uno de los investigadores y expertos más famosos en inteligencia artificial, define la inteligencia artificial como:

> "la ciencia de construir máquinas que hacen cosas que realizadas por el hombre requieren el uso de la inteligencia".

Otro punto de vista lo ofrecen B. G. Buchanan y E. A. Feigenbaum, de la Universidad de Stanford, cuando dicen que:

> "la investigación sobre IA es la parte de la ciencia de los ordenadores que investiga procesos simbólicos, razonamientos no algorítmicos y representaciones simbólicas del conocimiento usados en máquinas inteligentes".

Es importante recordar que se trata de un uso de los ordenadores ("máquinas inteligentes") que no respeta la habitual caracterización de un programa como la conjunción de unos algoritmos actuando sobre unas estructuras de datos. En realidad los programas de la IA son también, y tal vez esencialmente, no algorítmicos y así lo constataremos más adelante.

También, en una posición intermedia, P. H. Winston, director del laboratorio de inteligencia artificial del MIT, señala que:

> "el objetivo de la IA se puede concretar en cómo conseguir hacer ordenadores más útiles para comprender los principios que hacen posible la inteligencia".

En otras definiciones se intenta analizar el contenido semántico de los términos "inteligencia" y "artificial" que componen la denominación, pero este enfoque también parece plantear graves problemas por las dificultades de definir los mismos conceptos de inteligencia o de artificialidad.

Al lado del contenido ya práctico de la definición de Minsky, otros autores no dudan en proponer contenidos todavía más pragmáticos y dicen que:

> "la inteligencia artificial es el estudio de las facultades mentales mediante el uso de modelos computacionales"

Esta definición pone énfasis en el interés de las técnicas de la IA en duplicar (reproducir) las facultades mentales de las personas: visión, lenguaje natural, razonamiento y comprensión, por poner ejemplos concretos ya tratados por las técnicas de la IA.

Otra definición

Otra definición complementaria y en la misma línea, también procedente de un libro de texto, es la que ofrece Elaine Rich en 1983 al decir que "la inteligencia artificial es el estudio de cómo se puede hacer que los ordenadores hagan cosas que, de momento, las personas hacen mejor".

Quizá estas definiciones ilustran más exactamente el contenido actual de la inteligencia artificial que el ambicioso objetivo de superar el test de Turing con una máquina o un programa de ordenador.

1.4. Dos enfoques opuestos

Un elemento importante en la realidad actual e histórica de la inteligencia artificial es que hay dos enfoques opuestos sobre cómo se ha de conseguir este estudio y duplicación tecnológica de la inteligencia y los procesos cognoscitivos habituales en el ser humano.

Por una parte, **John McCarthy**, inventor de la denominación "inteligencia artificial", creador del lenguaje Lisp y uno de los investigadores más famosos de la IA, recomienda una línea de investigación que persigue la obtención de programas de ordenador que razonen siguiendo básicamente las exigencias de los dictados de la **lógica matemática**.

En el extremo opuesto, **Marvin Minsky**, también famoso experto de la IA, propone imitar fundamentalmente el método de **razonamiento de la mente humana** que, según él, no tiene por qué ser necesariamente el mismo de la lógica matemática.

Quizá el enfoque de McCarthy presenta más facilidades para llegar, cuando menos, a ciertos resultados. Pero la observación de Minsky establece las bases de una duda razonable sobre si los resultados obtenidos por el enfoque propuesto por McCarthy tienen realmente algo que ver con la inteligencia humana o tan sólo son meras aproximaciones que pueden cerrar el camino a la verdadera imitación de la inteligencia humana, si éste es el objetivo principal, lo cual, como ya se ha dicho, es puesto en duda por otros investigadores más pragmáticos.

De aquí que, en conjunto, las técnicas de la IA, en su intento de imitar las características de la mente humana, traten, en cualquier caso, el estudio de los métodos basados en la representación simbólica del conocimiento o en la inferencia simbólica, y se preocupen por problemáticas como la comprensión de los procesos cognoscitivos, el razonamiento y la toma de decisiones, la comprensión y la utilización del lenguaje natural –en oposición a lenguajes más formales y más "fáciles" para los sistemas informáticos–, la percepción visual, los procesos de aprendizaje, etc.

1.5. Otras denominaciones

El nuevo apogeo de las técnicas de la inteligencia artificial en los últimos años provocó, en la década de los ochenta, que algunos intentaran utilizar las mismas iniciales (IA) como indicadores también de *informática adelantada*, en un intento de expresar que, a pesar de sus más de treinta años de antigüedad, el éxito de las técnicas de la IA la configuraba como uno de los aspectos más

prometedores de la informática y uno de los más importantes con vista al futuro inmediato de la tecnología informática. Esta denominación, aunque ha sido frecuentemente utilizada en el mundo comercial y empresarial, resulta bastante despreciada en el mundo académico, preocupado básicamente por los aspectos teóricos de la IA y que, por lo tanto, recomienda preferentemente denominaciones como **sistemas basados en el conocimiento** o **ciencia de los procesos cognoscitivos**.

También hay que mencionar la observación de **John Haugeland**, quien propone el término **inteligencia sintética** para destacar, al lado de la artificialidad de la inteligencia conseguida por la IA, el hecho de que ésta resulta originada en la actividad humana, que "sintetiza" así una nueva forma de su propia inteligencia.

2. Aplicaciones de la IA

En la actualidad, las técnicas de la inteligencia artificial se aplican en muchos campos. A continuación destacaremos algunos de ellos:

- Sistemas expertos.
- Sistemas de visión por ordenador.
- Sistemas de lenguaje natural.
- Resolución de problemas.
- Robótica.
- Aprendizaje.
- Juegos y puzzles.

2.1. Sistemas expertos

Los sistemas expertos constituyen el eje central de una nueva actividad informática conocida como **ingeniería del conocimiento**. Con ellos se persigue la construcción de programas que reproduzcan de manera correcta el comportamiento de un experto humano en su dominio concreto de competencia y pericia.

Los sistemas expertos también reciben el nombre de *sistemas basados en el conocimiento* y, con intenciones más teóricas, *sistemas de producción*.

2.2. Sistemas de visión por ordenador

Después del objetivo central del antiguo *perceptron* de Rosenblat, los sistemas de visión por ordenador persiguen el **reconocimiento de formas** con el objetivo de conseguir la **visión por ordenador**, tanto con la finalidad de aplicarla en el ámbito de la robótica como en su aplicación en medicina, por ejemplo para ayudar a las personas que sufren ceguera.

***Perceptron* de Rosenblat**

Este tema se trata más adelante en el apartado "El *perceptron* de Rosenblat"

2.3. Sistemas de lenguaje natural

El lenguaje que utilizamos los seres humanos se ha etiquetado de "lenguaje natural" en contraposición a los lenguajes "artificiales", que se han diseñado para los ordenadores. La **comprensión del lenguaje natural** es muy difícil, ya que hay términos o expresiones con más de un significado.

Temas de estudio

Entre los **temas de estudio** del lenguaje natural se incluyen la comprensión y el análisis de textos, la traducción automática y la generación de informes o textos. También se incluyen el tratamiento de la palabra hablada y otras técnicas auxiliares como el reconocimiento vocal y el tratamiento y la comprensión de la expresión oral.

El tratamiento del lenguaje natural es uno de los temas más interesantes que trata la IA, tanto por el objetivo del estudio en sí mismo como por la posibilidad de poder acceder en un futuro a los ordenadores utilizando el mismo lenguaje natural que utilizamos los seres humanos, propósito central del proyecto japonés de la llamada "quinta generación de ordenadores".

2.4. Resolución de problemas

Uno de los temas centrales de la IA es el uso de la inteligencia para la resolución de todo tipo de problemas.

Los objetivos centrales han sido encontrar los mecanismos de deducción, buscar soluciones y conseguir la explicación de la solución que se ha obtenido y su justificación.

Resolutor general de problemas

Después del intento del resolutor general de problemas (GPS, *general problem solver*), en 1972 Newell y Simon publicaron un influyente libro sobre este tema: *Human Problem Solving*.

También se estudia la planificación inteligente de conductas y procedimientos de acción. Aunque en este tema los sistemas expertos han avanzado mucho más en sus aplicaciones puntuales, el estudio de la resolución de problemas continúa siendo la base teórica de un planteamiento más general.

2.5. Robótica

Como resultado de la evolución directa de la ingeniería mecánica y la automatización, la IA también intenta conseguir el **aumento de la habilidad** y la **autonomía** de estos mecanismos que manipulan instrumentos y que denominamos *robots*.

La robótica necesita tanto la percepción (visual o de otra manera) del entorno donde trabaja el robot, como la planificación de la acción que se ha de emprender en cada caso. Aunque con gran éxito y una aplicación creciente en el mundo industrial, los robots que se han conseguido crear todavía están muy lejos de los de ficción, que son los que configuran la imagen y las expectativas de la robótica en la mayoría del público.

2.6. Aprendizaje

Es posible que **la esencia de la inteligencia sea la capacidad de aprender**, es decir, de obtener conocimientos nuevos a partir de la experiencia y los conocimientos ya adquiridos. El objetivo de hacer que los programas "aprendan" es central en las técnicas de la IA. Uno de los ejemplos más conocidos son los programas que juegan a ciertos juegos de contenido estratégico (ajedrez, damas, bridge, etc.) en los que, frecuentemente, también se persigue que, a partir de los resultados obtenidos, mejoren su comportamiento en futuras partidas.

Al lado del interés que los programas de ordenador puedan "aprender", también se da mucha importancia al hecho de que la informática se pueda utilizar para favorecer el proceso de aprendizaje de los seres humanos. Se habla así **de enseñanza asistida por ordenador** (EAO), en la cual destaca, entre otras utilizaciones, la creación de las llamadas *situaciones de descubrimiento* gracias a la experimentación directa con ciertas herramientas informáticas.

Un elemento central de esta utilización pedagógica de la IA es el lenguaje Logo, desarrollado en el MIT por **Seymour Papert** adaptando ideas nacidas en el Lisp. El amplio universo pedagógico que ha abierto el Logo y sus "micromundos" es uno de los escasos ejemplos en los que la utilización de una nueva tecnología en la educación también ha venido acompañada de un método pedagógico *ad hoc*.

2.7. Juegos y puzzles

Una de las aplicaciones más inmediatas de la investigación en IA ha sido la elaboración de **programas de juegos** que, al mismo tiempo, puedan ser ejemplos de la aplicación de las técnicas de resolución de problemas y de aprendizaje, cuando se aplican a universos o dominios restringidos como son, en definitiva, los dominios que describen las reglas de los juegos. En este aspecto destaca la aplicación de técnicas de IA en programas de ordenador que juegan a damas, ajedrez o bridge y otros juegos estratégicos de sala para cuya práctica se presume una cierta necesidad de inteligencia.

Se han estudiado los juegos y los puzzles porque permiten tratar de manera restringida las dificultades generales de la resolución de problemas. Básicamente se intenta encontrar una representación adecuada del espacio de casos propios del problema para configurar lo que se denomina "espacios de estados".

Ejemplos de juegos

Ejemplos utilizados sobradamente en la investigación son los juegos sencillos como el tres en ralla, los puzzles simples, o disponer ocho reinas en un tablero de ajedrez sin que se "coman" entre sí.

La reducción de posibilidades de los universos explorados en los juegos –respecto del mundo real– ofrece la posibilidad de tratar de modo particular los graves problemas derivados de la explosión combinatoria del número de casos posibles y de las soluciones que hay que probar.

3. Técnicas de la IA

Cuando se aborda la resolución de un problema con técnicas de IA, se trata tanto de obtener la solución como de poder exponer el proceso de razonamiento que se ha utilizado. Obviamente, se deberán obtener varias soluciones si existe más de una y justificar el interés de cada una de ellas. También es importante que el sistema de IA sepa mejorar el razonamiento utilizado a medida que aumenta la experiencia, hecho que constituye su forma concreta de aprendizaje.

3.1. Técnicas de base de la IA

De la misma manera que la informática tradicional se compone de algoritmos que operan sobre determinadas estructuras de datos, la **técnica propia de la IA es obtener una adecuada representación simbólica del conocimiento**.

También es posible aplicar tanto los mecanismos de la **inferencia simbólica**, propios del **razonamiento deductivo**, como los mecanismos de tipo empírico que forman la base de la busca **heurística** de soluciones.

3.2. Inferencia simbólica

Los mecanismos de deducción típicos de la inferencia simbólica incluyen la **deducción** –también denominada a veces *inferencia lógicamente correcta*– amparada en las distintas reglas de la inferencia lógica:

a) El ***modus ponens***, con reglas del tipo: "si p entonces q", es decir, del hecho o premisa p se puede afirmar el hecho o premisa q.

b) La **instanciación universal**, que nos indica que si alguna cosa es cierta de todos los elementos de un conjunto, también es cierta para cada caso particular, como recuerda el conocido razonamiento: "todos los hombres son mortales, Sócrates es un hombre, por lo tanto, Sócrates es mortal".

Pero la IA también utiliza para sus razonamientos otros tipos de reglas de inferencia lógica, no tan inmediatas ni siempre seguras.

c) Un ejemplo de estos es el mecanismo "razonador" de la **inducción** por el cual, predicada una propiedad de una amplia serie de individuos, se "induce" que la propiedad puede ser predicada de todos los individuos de la misma especie.

d) Otro ejemplo de esto es una versión particular del llamado ***modus tollens*** de la lógica en la llamada abducción, que a partir de la regla "si *a* entonces *b*" y el hecho o premisa *b* se atreve a postular la posibilidad de *a*. Aunque la abducción pueda parecer una manera de proceder extraña y arriesgada, tiene su utilidad en la tentativa de busca de soluciones.

La diagnosis médica

Un ejemplo concreto de esto es la diagnosis médica, que viene a ser, en realidad, un uso prudente de la abducción, ya que, a partir de varios síntomas, el médico puede hacer una abducción de la existencia de una causa posible.

3.3. Heurística

Pero los mecanismos de la inferencia simbólica no son suficientes, y por ello los sistemas de IA utilizan muy a menudo las técnicas de la heurística.

La heurística se podría definir como el conjunto de los criterios, métodos o principios que se utilizan para encontrar, de entre varios caminos posibles, cuál o cuáles son los más efectivos para obtener un objetivo determinado.

Evidentemente, la heurística también tiene mucho que ver con los **mecanismos experimentales y empíricos**, cuya síntesis acaba elaborando estas reglas de la experiencia que se utilizan para seleccionar un camino de acción ante la explosión combinatoria de los muchos casos posibles.

Ejemplo

Investigaciones de IA han demostrado que un jugador de ajedrez experto, no analiza *todas* las jugadas posibles en un momento determinado del juego, sino tan sólo aquellas que, heurísticamente, reconoce, gracias a su experiencia, como más prometedoras o interesantes.

4. La representación del conocimiento

Uno de los temas centrales en la inteligencia artificial moderna es el de los diferentes sistemas que la IA utiliza para la representación del conocimiento e, implícitamente, las posibilidades que la mencionada representación ofrece para su utilización.

Lo que se persigue no es sólo unas estructuras de datos que proporcionen un sistema de representación efectivo y eficiente de los conocimientos, sino también qué conocimiento se ha de representar en cada utilización particular.

Los problemas anexos e importantes son:

1) La **extracción del conocimiento** de quien lo posee.

2) Su **formalización** en un determinado sistema de representación.

3) La posibilidad de crear el conocimiento mencionado, si se da el caso, o **modificar el conocimiento existente** gracias a la interacción del sistema de IA con el entorno en un mundo cambiante.

Obviamente, cada sistema de representación del conocimiento puede ir asociado a una determinada forma de razonar con el conocimiento almacenado, de manera que este conocimiento se utilice adecuadamente para obtener la solución deseada. Un problema quizá posterior, pero no menos importante, es la implementación informática de la representación del conocimiento elegida de manera que sea eficiente.

4.1. Componentes y elementos que se deben representar

En cualquier sistema de representación del conocimiento son componentes esenciales tanto las estructuras de datos que se utilizan, como sus procedimientos de interpretación y manejo.

Los distintos elementos que se han de representar son:

a) El conocimiento sobre los **objetos**.

b) El conocimiento sobre los **procesos**.

c) El conocimiento sobre el **entorno** en el que existen objetos y procesos.

Además, también se ha de implementar la representación de distintos objetivos, motivaciones, elementos de causalidad, temporalidad, etc. Uno de los elementos más difíciles de incorporar en los sistemas de IA es el aparentemente sencillo **sentido común**, que siempre supone un conocimiento difícil de determinar y precisar y que, a menudo, se acaba convirtiendo en reglas de tipo heurístico.

4.2. Etapas en el uso del conocimiento

En el uso del conocimiento se reconocen varias etapas:

a) La **adquisición**. Se trata de acumular nuevo conocimiento y relacionarlo con el que ya se tenía.

b) La **recuperación**. Se trata de obtener, de una amplia base de conocimientos, precisamente aquel que sea verdaderamente relevante para resolver el problema en curso.

c) El **razonamiento**. Se trata de inferir alguna cosa, a partir del conocimiento ya seleccionado, para obtener y verificar hechos nuevos diferentes de los ya conocidos.

4.3. Propiedades de un buen formalismo de representación del conocimiento

A un sistema de representación del conocimiento se le suelen exigir varias propiedades como:

a) La **adecuación representacional** o posibilidad de representar todas las clases de conocimiento que sean necesarias en un dominio dado.

b) La **adecuación inferencial** o posibilidad de manipulación de las estructuras de datos que sirven para representar el conocimiento, de manera que sea posible generar nuevas estructuras que correspondan a nuevos conocimientos inferidos de los anteriores.

c) La **eficiencia inferencial** o capacidad de incorporar también un nuevo conocimiento adicional (*metaconocimiento*) que puede utilizarse para mejorar el uso de los mecanismos de inferencia y optimizar el funcionamiento del sistema de IA.

d) La **eficiencia en la adquisición del conocimiento** o facilidad de adquirir información nueva incluyendo el hecho de que, en el caso ideal, el propio sistema de IA debe ser capaz de controlar la adquisición de nuevo conocimiento o metaconocimiento.

4.4. Los espacios de estados

Uno de los primeros formalismos utilizados para la representación del conocimiento fue el denominado **espacio de estados**, que describe en sus **estados** todos los momentos y pasos tanto del problema, como de la solución. Esta solución se obtiene con unos operadores o **transiciones** que actúan transformando la situación del problema de un estado a otro dentro del espacio de estados.

Este sistema resulta adecuado cuando no se produce una explosión combinatoria del número de casos posibles, lo cual puede darse cuando se abordan universos reducidos como sucede con algunos juegos o puzzles. Pero en otros casos, como en el conocido juego del ajedrez, la explosión combinatoria de estados es tal que resulta imprescindible recurrir a reglas de tipo heurístico para simplificar la multitud de estados posibles.

4.5. Esquemas modernos de representación del conocimiento

El agotamiento de la capacidad de representación y manipulación de los espacios de estados hizo aparecer otros esquemas de representación del conocimiento.

Se suele distinguir entre los **esquemas de tipo declarativo**, como los esquemas lógicos y las redes semánticas, y los **esquemas de representación de procedimientos**, como los sistemas de representación procedural o los sistemas de producción, también denominados *sistemas expertos*. Destaca también la tendencia integradora representada por los ***frames*** y los ***scripts***.

4.5.1. Los esquemas lógicos

En los esquemas lógicos (un ejemplo sencillo de esquemas de tipo declarativo) los hechos y las reglas son fórmulas expresadas en algún sistema basado en la lógica, como sucede con el lenguaje Prolog, que utiliza la lógica de primer orden del cálculo de predicados. También existe el uso de lógicas más complejas como la lógica multievaluada, la temporal o la difusa, por mencionar ejemplos ya utilizados en la IA.

Una **fórmula lógica** es, en definitiva, una combinación de predicados, variables, conectores lógicos, cuantificadores y funciones.

La programación lógica

El ejemplo más clásico del uso de esquemas lógicos es la programación lógica implementada con el lenguaje Prolog y su uso de los mecanismos de resolución, unificación y resolución por refutación.

La ventaja de este sistema es que se trata de un procedimiento preciso, flexible, modular y bastante natural, aunque suele carecer de principios de organización y selección entre los hechos (deficiencias del metaconocimiento). También suele tratar objetos demasiado simples y resulta difícil de manipular si hay muchos datos.

4.5.2. Las redes semánticas

Las redes semánticas son unos sistemas de representación, inicialmente gráficos, que resultan muy adecuados para establecer taxonomías. Al grafo que constituye una red semántica, los **nodos** incorporan objetos o conceptos y los **arcos** son relaciones entre los nodos.

En los nodos se describen tanto **conceptos** –constantes o parámetros de la realidad que especifican objetos o abstracciones–, como **acontecimientos** –acciones que ocurren en la realidad que se está modelando– y **características** –que particularizan estados, modifican conceptos, acontecimientos y también otras características.

En los arcos se suelen distinguir los que sirven a efectos de clasificación, que relacionan un objeto con otros de su clase como sucede con los casos "miembro-de" o "instancia-de", de **agregación**, que relacionan un objeto con sus componentes con mecanismos como "parte-de", y de **generalización**, que relacionan un objeto con una clase más general con ejemplos del tipo "es-un" o "es-subconjunto-de".

Variantes evolucionadas

Algunas variantes evolucionadas a partir de las redes semánticas son las redes particionadas (como propone Hendrix), los esquemas de propagación de *markers* (propuesta de Fahlman), las jerarquías de tópicos (del tipo "es-un" o "parte-de"), las redes proposicionales (como sugiere Shapiro), etc.

La característica fundamental de las redes semánticas es su facilidad para implementar la **herencia de propiedades**, lo que permite deducir hechos nuevos derivados del hecho de que los nodos más altos en la jerarquía son también aserciones sobre los nodos más bajos.

En las figuras que podéis ver al margen se muestran dos ejemplos concretos de las muchas formas que puede adoptar una red semántica.

Formas que puede adoptar una red semántica

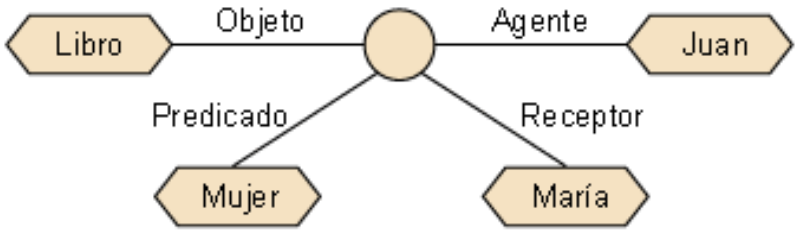


Figura 1. Juan da un libro a María

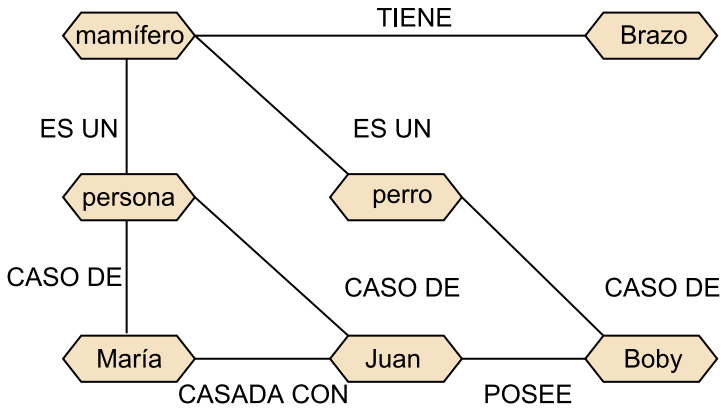


Figura 2. Una jerarquía "ES-UN"

4.5.3. Los esquemas de representación de procedimientos

Los sistemas de representación procedural se basan en la idea de almacenar el conocimiento en forma de procedimientos (*procedures*) en lugar de hacerlo en forma de proposiciones.

Pero parece que se ha obtenido mejor resultado con lo que los teóricos denominan sistemas de producción y que, más frecuentemente, reciben el nombre de sistemas expertos, que estudiaremos con más detalle en el apartado 5.

En este caso se trata de:

a) Un conjunto de reglas de producción que constituye la base de conocimientos.

b) Una base de hechos que describe el contexto de utilización del sistema experto.

c) Un intérprete o motor de inferencia.

El motor de inferencia obtiene un nuevo conocimiento por un procedimiento denominado de unificación –selección de la regla que se ha de utilizar en cada caso–, una resolución de los posibles conflictos entre reglas y una acción que decide reglas o hechos nuevos que se han de almacenar en la base de hechos.

4.5.4. Los *frames*

El ***frame***, que los expertos españoles han decidido no traducir por "marco", se puede asimilar a una red semántica compleja de estructuras que se utilizan para describir la colección de atributos que tiene un determinado objeto o una situación compleja. Compartiendo las propiedades de una red se puede esta-

Información en los frames

Propuestos por Minsky en un artículo hoy clásico, "A Framework for representing Knowledge" (1975), los *frames* incluyen en sus estructuras de datos tanto información procedural, como declarativa.

blecer una jerarquía de *frames*, también apta para establecer taxonomías tanto de pertenencia a una clase (*member link*, del tipo "Juan ES-UN hombre"), como de adscripción a otra (*subclass link*, del tipo "el hombre ES-UN mamífero").

Los *frames* disponen de **ranuras de expansión (*slots*)** como lugares físicos para insertar una parte de conocimiento o propiedad dentro de la estructura. Existen las ranuras de expansión **miembro**, que describen los atributos de cada miembro de una clase y se utilizan sólo en los *frames* que representan clases, y las ranuras de expansión propias, que describen los atributos de la clase en sí.

La ventaja de los *frames* sobre las redes semánticas es que también contienen información procedural. Esta información está en los **métodos**, que vienen a ser procedimientos escritos, por ejemplo en lenguajes como el Lisp, y que se almacenan como valores de las ranuras de expansión.

También utilizan los **valores activos (*demons*)** como procedimientos o reglas de producción ligados a una ranura de expansión y que se utilizan cuando cambia la información que contiene, tanto a la hora de añadir nueva como a la de eliminar o modificar la que ya contiene.

Además, los *frames* incorporan métodos de inferencia como la herencia de propiedades, propia de las redes semánticas y también el razonamiento mediante *facets*, que son restricciones sobre el número de valores posibles que puede tener un atributo y sobre las clases a las que puede pertenecer cada valor.

4.5.5. Los *scripts*

Un último ejemplo de sistema de representación mixta es el de los ***scripts***. Se trata de estructuras especializadas que describen secuencias de acontecimientos en un contexto particular. Los componentes principales de un *script* son:

a) Las **condiciones de entrada** que se han de satisfacer antes de que los acontecimientos descritos en el *script* se puedan iniciar.

b) El **resultado** o las condiciones que sean ciertas cuando los acontecimientos descritos se hayan cumplido.

c) Las **propiedades** (*prop*), que son ranuras de expansión que representan los objetos involucrados en los acontecimientos que describe el *script*.

d) Los **personajes** (*roles*), que son las diferentes entidades o personas que intervienen en los acontecimientos descritos.

e) La **instancia** (*track*), que es una variación específica o un caso particular de un modelo más general que se ha representado mediante este *script*.

f) Las **escenas**, que son las diferentes secuencias de acontecimientos que tienen lugar en el *script*.

5. Sistemas expertos

5.1. Definición y características

Los sistemas expertos, también denominados sistemas basados en el conocimiento, representan uno de los éxitos más importantes de la IA, al menos con respecto al éxito de sus aplicaciones en el ámbito de los sistemas de información.

La finalidad principal de los sistemas expertos es la reproducción correcta del comportamiento de un experto humano en su dominio de competencia.

Los sistemas expertos

Son uno de los ejemplos más claros de la evolución de la IA y de su éxito cuando ésta ha prescindido de los objetivos exageradamente ambiciosos de los primeros años, y ha delimitado el dominio y las características de los problemas que se han de resolver.

Algún especialista ha querido etiquetar los sistemas expertos como "la IA con éxito", algo que es, evidentemente, una exageración, pero que indica la importancia de los sistemas expertos en la superación de los "años difíciles" de la IA, a principios de los años sesenta, con varios proyectos en curso y un número excesivo de fracasos.

5.1.1. Atributos de un experto humano

Un experto humano es aceptado como tal por una serie de atributos que caracterizan su tarea, en la cual se le reconoce una verdadera competencia, habilidad y experiencia. En general, el experto humano que aquí nos interesa considerar resuelve problemas complejos, actualiza sus conocimientos y es capaz de justificar sus decisiones y respuestas, a las cuales también puede atribuir grados de credibilidad.

El buen experto humano también es capaz de destacar los aspectos más relevantes de un problema, a la vez que detecta y reconoce la desaparición de su competencia o pericia cuando el problema en cuestión se acerca a las fronteras del dominio de su experiencia, es decir, es consciente de los límites de su competencia.

Asimismo es capaz de aprender a razonar mejor, reestructurando sus conocimientos, aplicando procedimientos o reglas para los casos de excepción y elaborando nuevas reglas o procedimientos para enfrentarse a nuevos problemas. Por si no es suficiente con todo eso, además dispone de sentido común para situar su manera de proceder dentro de un ámbito más general.

5.1.2. Definición y características de un sistema experto

Un sistema experto es un sistema elaborado con técnicas de IA que, de la misma manera que el experto humano a quien intenta emular, resuelve problemas complejos y difíciles que se circunscriben a un dominio específico y delimitado.

El sistema experto utiliza procesos que imitan el razonamiento humano (deducción, inducción, estrategias de busca de soluciones, etc.) a la hora de resolver los problemas. Y todo esto lo consigue utilizando los conocimientos de base, suministrados en origen por un experto humano, a los que también incorpora los conocimientos que el sistema experto informático "aprende" durante su actividad como "experto".

Además, el sistema experto ha de ser capaz de justificar las decisiones y los resultados obtenidos y atribuirles grados de credibilidad. También, como hacen a veces a los expertos humanos, ha de ser capaz de razonar a partir de datos inciertos.

5.2. Estructura básica de un sistema experto

En la figura siguiente se muestra la estructura básica de un sistema experto en la que se percibe claramente un aspecto fundamental: la separación de los datos que forman el conocimiento, de las estrategias de resolución o procedimiento con las que se elaboran las soluciones.

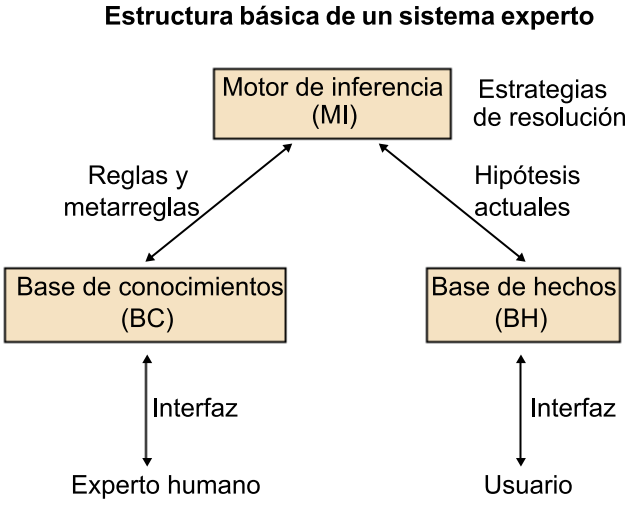


Figura 3

Los conocimientos se almacenan en la **base de conocimientos** y en la **base de hechos**, y los procedimientos capaces de razonar se implementan en el **motor de inferencia**.

El esquema también muestra la necesidad de interfaces que permitan el acceso al sistema, tanto del experto humano que alimenta la base de conocimientos como del usuario del sistema experto que proporciona los hechos que determinan una utilización concreta del sistema experto. El usuario también obtiene las respuestas que proporciona el sistema.

5.2.1. La base de conocimientos

La base de conocimientos viene a ser la **memoria a largo plazo** del sistema experto y contiene las **reglas** y las **metarreglas** que resumen el conocimiento del experto humano sobre el dominio del problema. Las metarreglas indican el orden o la preferencia con la que se deben usar las reglas que describen el conocimiento del experto en cada caso.

Una de las formas habituales (y también más simples) es la utilización de sistemas sencillos de representación del conocimiento, como son las **reglas lógicas** del tipo "si ... entonces...".

Las reglas son actualizables (y utilizables) de una en una, e independientes de su uso posterior.

Ejemplo 1

SI (la persona-X ES-UN hombre)

Y (la persona-Y ES-HIJO de la persona-X)
Y (la persona-Z ES-HIJO de la persona-Y)

ENTONCES (la persona X ES-ABUELO de la persona-Z).

Ejemplo 2

SI (el país-X y el país-Y SON-VECINOS)

ENTONCES (el color del país-X ES-DIFERENTE del color del país-Y).

5.2.2. La base de hechos

La base de hechos, también denominada a veces **memoria a corto plazo** o **memoria de trabajo**, almacena tanto los hechos proporcionados por el usuario, y que describen la situación concreta analizada, como los hechos nuevos que el propio sistema experto va obteniendo.

Ejemplo 1

Juan ES-UN hombre

Y ES-HIJO de Carlos.

Ejemplo 2

El amarillo ES-UN color.

5.2.3. El motor de inferencia

El motor de inferencia contiene el **mecanismo de razonamiento** que sigue el sistema experto: deducción, inducción, estrategias de busca de soluciones, etc.

> Con ello, el motor de inferencia, al aplicar las reglas contenidas en la base de conocimientos sobre la base de hechos, obtiene nuevo conocimiento que se incorpora también a la base de hechos y es objeto, a su vez, de la aplicación de las reglas recogidas en la base de conocimientos.

Uno de los problemas centrales en el funcionamiento del motor de inferencia cuando las bases de conocimientos y de hechos son voluminosas es la selección adecuada de las reglas y los hechos que se han de considerar en cada paso del "razonamiento". En este aspecto es fundamental la existencia de **metarreglas** específicas que orienten precisamente la selección mencionada.

5.2.4. Las interfaces

Un elemento esencial en la facilidad de uso del sistema experto son las interfaces de entrada de datos que permiten la interacción, tanto del experto como del usuario, con el sistema experto. También existe la posibilidad, no recogida en el esquema de la figura 4 que repetimos al margen, de incorporar una interfaz, realizada con los sistemas de la informática tradicional, que permita la extracción de datos de la base de datos de gestión para alimentar parte de la base de hechos. En las interfaces con los usuarios humanos también es habitual buscar las formas más amigables del diálogo persona-máquina.

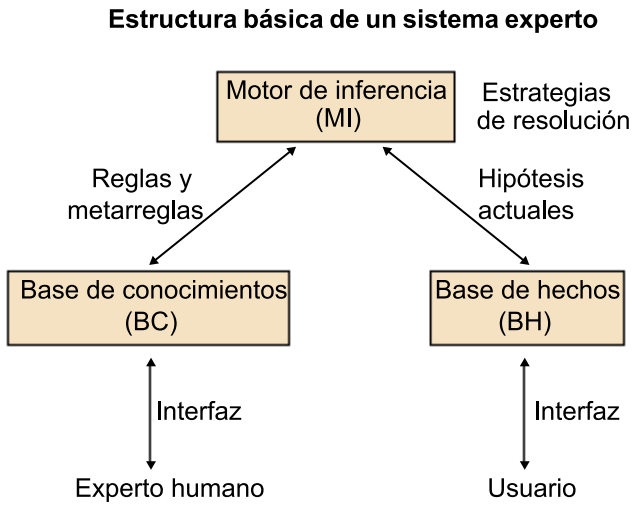


Figura 4

La figura 4, al margen, muestra el entorno de desarrollo en el que el experto humano, quizá ayudado por un ingeniero del conocimiento, dialoga con el sistema para crear la base de conocimientos.

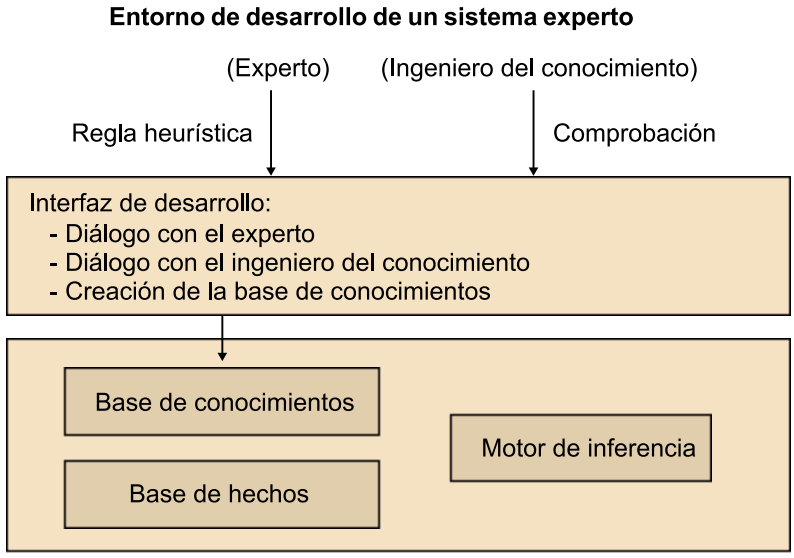


Figura 5

Asimismo, la figura 5, situada también al margen, muestra un posible entorno de utilización del sistema experto, en el cual un usuario dialoga con el sistema para crear la base de hechos e interrogar al sistema experto para obtener la justificación de las recomendaciones o decisiones de éste (proporcionadas por medio de la actividad "razonadora" y "explicadora" del motor de inferencia).

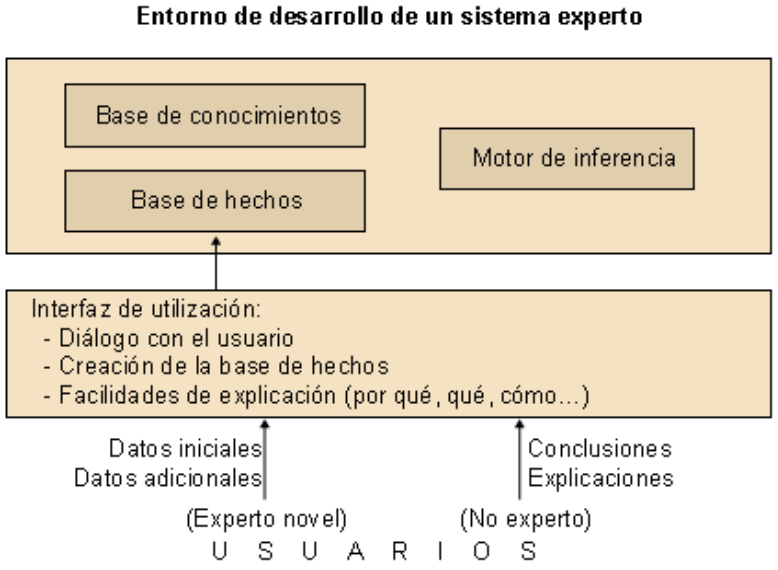


Figura 6

5.3. Bases de conocimientos: tipos de reglas

Incluso en el caso sencillo de conocimientos almacenados en forma de reglas simples de la lógica elemental, hay que considerar casos como los siguientes.

5.3.1. Reglas antecedente-consecuencia

Son del tipo *si todos los antecedentes son ciertos, entonces todas las consecuencias son ciertas* y responden al esquema general:

```
SI          <antecedente-1>
            <antecedente-2>
            ...
ENTONCES    <consecuencia-1>
            <consecuencia-2>
```

de lo que, en un sistema experto de diagnóstico médico, se obtendría el ejemplo siguiente:

```
SI          (paciente TIENE síntoma-1)
            Y (paciente TIENE síntoma-5)
            Y (paciente TIENE síntoma-7)
ENTONCES    (paciente TIENE enfermedad-A)
```

5.3.2. Reglas condición-acción

Son del tipo *si todas las condiciones son ciertas, entonces se deben ejecutar todas las acciones* y responden al esquema general:

```
SI          <condición-1>
            <condición-2>
            ...
ENTONCES    <acción-1>
            <acción-2>
            ...
```

de lo que, en un sistema experto para jugar al ajedrez, se obtendría el ejemplo siguiente:

```
SI          (torre-negra A B2)
            Y (reina-blanca A D2)
            Y (C2 ES LIBRE)
            Y (MUEVEN NEGRAS)
ENTONCES
            BORRAR:  (torre-negra A B2)
            BORRAR:  (reina-blanca A D2)
            BORRAR:  (MUEVEN-NEGRAS)
            INSERTAR:(torre-negra A D2)
            INSERTAR:(MUEVEN-BLANCAS)
```

5.3.3. Reglas inexactas

Son del tipo: *si todos los antecedentes son ciertos, entonces la consecuencia es cierta con un determinado grado de certeza* y responden al esquema general:

```
SI              <antecedente-1>
                <antecedente-2>
                ...
ENTONCES        <consecuencia-1>
CON UNA CERTIDUMBRE DEL <factor-de-certidumbre>
```

de lo que, en un sistema experto de diagnóstico médico, se obtendría el ejemplo siguiente:

```
SI              (paciente TIENE síntoma-2)
                Y (paciente TIENE síntoma-5)
                Y (paciente TIENE síntoma-8)
ENTONCES        (paciente TIENE enfermedad-B)
CON UNA CERTIDUMBRE DEL: 0,7
```

5.4. Motor de inferencia: funcionamiento y mecanismos de razonamiento

El motor de inferencia combina los hechos y las reglas para obtener nuevos hechos.

El funcionamiento típico de un motor de inferencia se ilustra en el ejemplo siguiente:

```
HECHO:  (paciente TIENE síntoma-3)
HECHO:  (paciente TIENE síntoma-7)
REGLA:  SI (paciente TIENE síntoma-3)
        Y (paciente TIENE síntoma-7)
        ENTONCES (paciente TIENE enfermedad-C)
```

de lo que el motor de inferencia puede obtener la conclusión siguiente:

```
HECHO INFERIDO: (paciente TIENE enfermedad-C)
```

5.4.1. Razonamiento por encadenamiento hacia delante

Una de las estrategias generales a disposición del motor de inferencia es el llamado encadenamiento hacia delante (*forward chaining*).

Se trata de un procedimiento gobernado por los datos (*data driven*) en el que se parte de los hechos conocidos y, con el uso de las reglas, se van deduciendo hechos nuevos.

Razonamiento por encadenamiento hacia delante (*forward chaining*)

En el ejemplo de la figura 7, los hechos iniciales (C y F) permiten obtener el nuevo hecho E al aplicar la regla 1 (*si* C *entonces* E). Una vez se dispone del hecho E, su combinación con uno de los hechos iniciales (F) permite que la regla 3 (*si* E y F *entonces* D) obtenga un nuevo hecho D en la base de hechos.

Ahora la aplicación de la regla 2 (*si* D *entonces* B) permite obtener un nuevo hecho B. Y finalmente la aplicación de la regla 4 (*si* B y C *entonces* A) permite completar la base de hechos con el nuevo descubrimiento A.

La figura 7 muestra en su parte derecha inferior la evolución de la memoria de trabajo (base de hechos) y las conclusiones finales obtenidas.

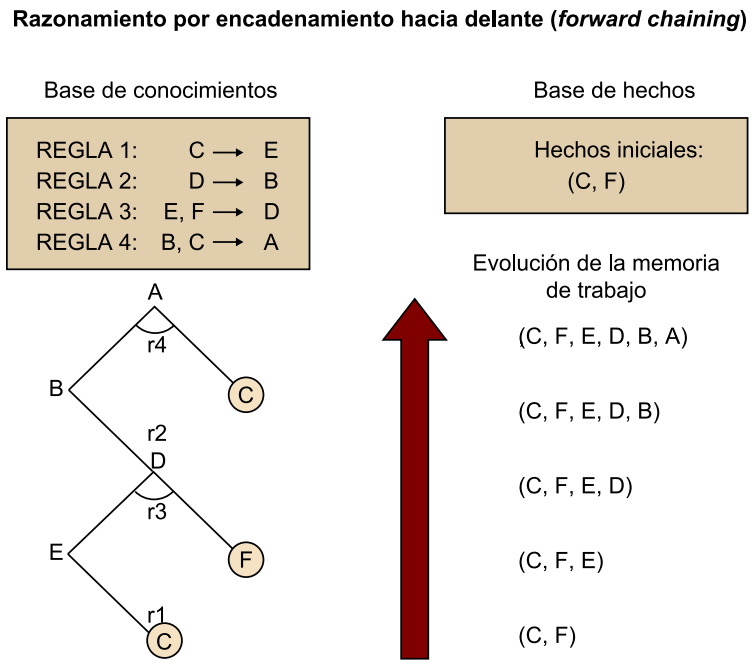


Figura 7

5.4.2. Razonamiento por encadenamiento hacia atrás

Otra estrategia habitual es el encadenamiento hacia atrás (*backward chaining*).

En el caso del encadenamiento hacia atrás (*backward chaining*), se trata de un procedimiento de busca dirigida por el intento de afirmar un objetivo concreto (*goal directed*) que se intenta probar de acuerdo con los hechos y las reglas disponibles.

Razonamiento por encadenamiento hacia atrás (*backward chaining*)

En el ejemplo de la figura 8, al intentar deducir si el objetivo A es cierto, la regla 4 (*si* B *y* C *entonces* A) permite sustituir el objetivo inicial A por el conjunto B y C.

Como C ya existe en la base de hechos, el objetivo A se cumplirá si se cumple el objetivo B. Con esto B se convierte en el nuevo objetivo que se debe verificar.

Para verificar B, la regla 2 (*si* D *entonces* B) permite sustituir el objetivo B por un nuevo objetivo D. Ahora la regla 3 (*si* E *y* F *entonces* D) dice que D se cumple si lo hacen E y F, y como F ya forma parte de los datos iniciales de la base de hechos, permite sustituir el objetivo D por un nuevo objetivo E.

Finalmente, la regla 1 (*si* C *entonces* E) hace ver que E se cumple si se cumple C. Como C forma parte de la base de hechos inicial, se cumple y, consiguientemente, también se cumplen todos los otros objetivos parciales (D y B) hasta llegar al objetivo inicial A, que queda así comprobado.

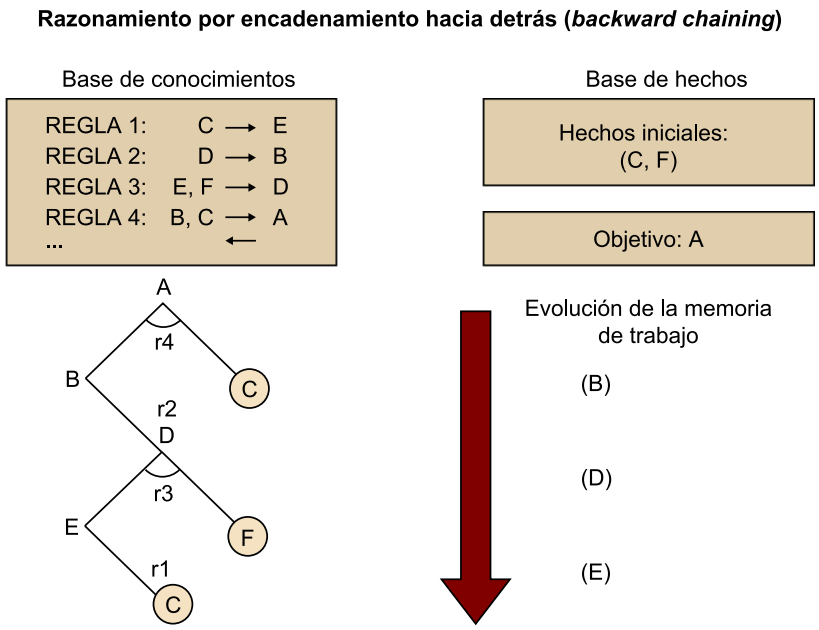


Figura 8

5.4.3. Razonamiento inexacto

Cuando las reglas pueden ser inexactas, el motor de inferencia ha de saber tratar con los factores de certeza (FC) para componer el factor de certeza de la solución propuesta. Esto se realiza utilizando reglas derivadas de la lógica difusa (*fuzzy logic*) que combinan tanto las leyes de la lógica, como las de la probabilidad.

Forma de composición de factores de certeza

En el ejemplo siguiente se muestra una posible forma de composición (producto) de factores de certeza (FC-entrada · FC-regla):

HECHO: A [FC = 0,8]
HECHO: B [FC = 0,7]
REGLA: Si A y B, entonces C [FC = 0,5]
Certidumbre de la entrada: mínimo 0,8 • 0,7 = 0,7
Certidumbre de la salida: 0,7 • 0,5 = 0,35

que permite establecer que C es cierto con un 35% de probabilidad, a partir del 80% de probabilidad asignada a la presencia del hecho A y el 70% de probabilidad asignada al hecho B. La entrada tiene, por lo tanto, una certeza del 70%, mientras que la regla aplicada sólo es cierta en el 50% de los casos. Por ello la certeza final queda reducida al 35%.

5.5. Ingeniería del conocimiento

Los sistemas expertos han llevado a la aparición de una nueva especialidad en la informática que se suele etiquetar de "ingeniería del conocimiento". De la misma manera que un administrador de bases de datos define la base de datos con un lenguaje de definición de datos (DDL), el ingeniero del conocimiento ayuda a definir la base de conocimientos con un sistema adecuado de representación del conocimiento. El paralelismo con el mundo de las bases de datos se completa con la figura del usuario del sistema experto, que actúa en cierto

modo como el usuario de un sistema de bases de datos que puede obtener, modificar y añadir nuevos datos puntuales con un lenguaje de manipulación de datos (DML). Igualmente, el usuario de un sistema experto añade nuevos hechos a la base de hechos y obtiene nuevos hechos deducidos de los que ya existen en la base.

La actividad del ingeniero del conocimiento supone la interacción con el experto humano, que es quien proporciona realmente el conocimiento que el ingeniero del conocimiento estructurará y ayude a introducir en la base de conocimientos. En algunos casos, si la interfaz de desarrollo es bastante amigable (*user friendly*): una vez decidido el formalismo de representación del conocimiento, el experto humano es quien alimenta directamente la base de conocimientos, aunque generalmente lo hace con la ayuda y supervisión del ingeniero del conocimiento.

5.5.1. Funciones del ingeniero del conocimiento

La actividad del ingeniero del conocimiento lo lleva a:

1) Descomponer un problema en subproblemas discretos.

2) Diseñar un esquema de representación del conocimiento para los hechos, las reglas y el dominio que se han de considerar.

3) Caracterizar las situaciones y las conclusiones del problema en términos del esquema de representación del conocimiento elegido.

4) Definir las reglas que permiten alcanzar soluciones cuando una situación particular llega a ser reconocida.

5) Seleccionar los estilos de razonamiento que se deben utilizar (encadenamiento hacia delante o hacia atrás, razonamiento inexacto, etc.).

6) Especificar la interfaz de utilización que se utilizará durante las actualizaciones y consultas que realice el usuario del sistema experto.

Para ello, el ingeniero del conocimiento ha de disponer de competencias propias de la **ingeniería de software**, es decir, la arquitectura de software de los sistemas expertos, los lenguajes que utiliza la IA (Lisp, Prolog, etc.) y las herramientas disponibles para la construcción del sistema experto.

El ingeniero del conocimiento también debe ser competente en lo que se denomina **ingeniería del conocimiento**, que implica básicamente conocer los sistemas de representación del conocimiento y las técnicas adecuadas, incluso de tipo psicológico, para extraer el mencionado conocimiento de los expertos humanos.

5.5.2. *Shells* y conjuntos de herramientas (*toolkits*)

Para el desarrollo efectivo de sistemas expertos es posible utilizar un **paquete adaptable**, es decir, una aplicación escrita previamente a la que realizar unos ajustes mínimos para adaptarla a unas necesidades concretas. En el extremo opuesto es posible un **desarrollo específico** utilizando los lenguajes propios de la IA (Lisp y Prolog generalmente) para construir un sistema con todos sus elementos y detalles: motor de inferencia, base de conocimientos e interfaces.

La dificultad y el coste del desarrollo específico se pueden evitar recurriendo a los paquetes generadores de sistemas expertos que se conocen habitualmente como *shells* o conjuntos de herramientas (*toolkits*).

En este caso se trata de un subsistema que incluye un motor de inferencia ya elaborado y, a menudo, un único esquema de representación del conocimiento, también prefijado. La construcción del sistema experto se reduce a obtener el conocimiento del experto humano –tarea que no es fácil ni banal– e introducirlo en el sistema gracias a interfaces de desarrollo también preexistentes y proporcionadas por la *shell* o conjunto de herramientas.

En la selección de una *shell* o conjunto de herramientas es importante conocer el esquema de representación del conocimiento que utiliza (reglas, *frames*, etc.), el mecanismo de razonamiento que incorpora su motor de inferencia (encadenamiento hacia delante o hacia atrás), el hardware en el que la *shell* está implementada (ordenadores personales, máquinas Lisp, miniordenadores, grandes sistemas, etc.) y el carácter abierto o cerrado de su arquitectura.

Otros elementos de interés son las facilidades de explicación de las decisiones obtenidas por el sistema experto, la posibilidad o no de razonamiento inexacto, el carácter de intérprete o compilador de la *shell*, principalmente en el momento de la adquisición del conocimiento, etc.

5.6. Viabilidad y beneficios de un sistema experto

Los sistemas expertos intentan emular y reemplazar a los expertos humanos y por ello son adecuados para su implantación en las organizaciones y situaciones en las que haya escasez de expertos o éstos resulten sometidos a las consultas de un amplio conjunto de usuarios muy dispersos geográficamente. También pueden suplir al experto humano en los casos en los que la formación de un experto nuevo sea muy costosa y el uso de su experiencia no sea bastante frecuente para justificarla.

Otras causas que aconsejan recurrir a la implementación de un sistema experto suelen ser el riesgo de la pérdida del experto humano y la existencia de un dominio en evolución constante.

Con respecto al problema abordado, deberá ser un problema que no tenga una solución algorítmica directa –si la tuviera, se trataría con las técnicas de la informática tradicional– y que requiera tareas deductivas e inductivas al lado de mecanismos heurísticos, en un dominio sobre el cual los conocimientos sean de tipo predominantemente cualitativo y simbólico, a menudo incompletos e imprecisos e, incluso, a veces, aparentemente contradictorios.

5.6.1. Requisitos de viabilidad de un sistema experto

Para que el sistema experto sea viable y pueda ofrecer resultados también es necesario que:

a) El dominio en el que actúa sea restringido y perfectamente definido.

b) Los casos (hechos, reglas y soluciones) sean claramente formalizables.

c) Las soluciones con un cierto grado de incertidumbre sean aceptables.

d) El problema sea de una complejidad "razonable" y su resolución por el experto no sea ni demasiado corta (caso banal) ni demasiado larga (caso demasiado complejo), lo cual se suele concretar, por ejemplo, en un número de reglas inferior al millar y una profundidad de inferencia de entre 2 y 50.

e) Existe disponibilidad de un experto humano, si es posible motivado, y con una cierta capacidad de formalización, ya que, en definitiva, es él quien proporciona el conocimiento.

> Esta disponibilidad de un experto humano motivado es un requisito primordial. Se puede decir que no puede haber un sistema experto sin un buen experto humano a quien emular.

5.6.2. Beneficios que se han de obtener de un sistema experto

Un buen sistema experto puede proporcionar muchos beneficios tanto para el experto, al cual teóricamente sustituye, como para los usuarios y las organizaciones que lo tienen a su disposición.

Para el **experto humano**, un sistema experto puede ser una herramienta de gran utilidad con respecto a formalización, estructuración y validación de sus propios conocimientos, ya que puede ver cómo estos conocimientos se aplican

a casos menos frecuentes y, quizá, con resultados insospechados. Gracias al sistema experto, el experto humano puede quedar reservado para las consultas menos rutinarias.

Para el **usuario**, un sistema experto implica la disponibilidad inmediata de la competencia del experto; una interactividad posiblemente más fácil si la interfaz de utilización está bien diseñada y se puede disponer de una justificación adecuada, tanto de las respuestas que obtiene del sistema experto como de las preguntas que el mismo sistema puede ir efectuando.

Para las **organizaciones** o empresas que utilizan un sistema experto, hay varias mejoras evidentes: la posible disponibilidad de un experto en cada punto de consulta, la perpetuación de la habilidad y competencia del experto y un aumento del nivel general de competencia de la organización.

También existe más facilidad para la formación de nuevos expertos humanos, que se pueden formar utilizando el sistema experto y comparando las recomendaciones con las que ellos mismos habrían realizado.

5.7. Problemas en el uso de sistemas expertos

Actualmente la realización de sistemas expertos todavía presenta serias limitaciones y problemas entre los cuales destacan:

a) La dificultad en la adquisición de los conocimientos obtenidos del experto humano, no siempre suficientemente colaborador y motivado.

b) Una aplicación efectiva que continúa siendo efectiva sólo en dominios todavía muy restringidos.

c) Un comportamiento muy frágil cerca de las fronteras del dominio, y una gran dificultad para que el sistema experto detecte el desbordamiento de su ámbito de competencia.

d) La obtención de explicaciones no siempre relevantes.

e) Las limitaciones serias por razones de eficiencia que obligan a restringir el número de reglas.

f) Las limitaciones implícitas a la potencialidad de las técnicas actuales de representación del conocimiento.

g) La escasa capacidad de los sistemas expertos actuales para el aprendizaje, la generalización y el razonamiento por analogía.

También es un problema la dificultad adicional de la integración de los sistemas expertos, y en general de los sistemas derivados de la IA, con la informática tradicional, hasta ahora gestora y depositaria de los "hechos".

5.8. Algunos sistemas expertos famosos

En este apartado mencionaremos algunos ejemplos concretos de sistemas expertos famosos:

- MYCIN.
- MACSYMA.
- PROSPECTOR.
- DENDRAL.
- XCON.

5.8.1. MYCIN

El MYCIN fue uno de los primeros sistemas expertos aparecido a mediados de los años setenta.

Lo desarrolló Shortliffe en la Universidad de Stanford y se ocupa del **diagnóstico de infecciones de la sangre y su terapia**. Utiliza reglas de tipo lógico, su motor de inferencia es por encadenamiento hacia atrás (*backward chaining*) y es capaz de utilizar factores de certeza.

Posteriormente se completó con **TEIRESIAS**, que es una versión "explicativa" del MYCIN que justifica el diagnóstico y la terapia recomendada.

5.8.2. MACSYMA

Derivado tardío del SAINT de Slage, desarrollado en el MIT a mediados de los años cincuenta, el MACSYMA, documentado en 1977, trata el **cálculo diferencial e integral** con un motor de inferencia compuesto de una serie de funciones implementadas directamente en Lisp.

5.8.3. PROSPECTOR

Documentado por **Hart** (1978) y **Duda** (1979), se ocupa de la **prospección y evaluación de yacimientos de minerales**, particularmente de cobre y uranio. Utiliza la inferencia probabilística.

5.8.4. DENDRAL

Desarrollado en la Universidad de Stanford bajo la dirección de Feigenbaum, es capaz de **analizar la estructura molecular de un compuesto a partir del espectograma de masas y otros datos**. Su motor de inferencia usa el mecanismo de *generate and test*, que consiste en generar casos posibles y probar si cumplen todas las condiciones.

5.8.5. XCON

El XCON es una versión operativa de un sistema anterior denominado *R1* y, posiblemente, el primer sistema experto utilizado en el ámbito comercial. Se utilizó, en la firma DEC, **para la configuración de los sistemas DEC/VAX** y lo desarrolló **J. McDermott** de la Universidad de Carnegie-Mellon. Utiliza un motor de inferencia con encadenamiento hacia delante (*forward chaining*) y es operativo desde 1981.

5.8.6. Otras aplicaciones

El campo de aplicación posible de los sistemas expertos es realmente amplio. A modo de ejemplos no exhaustivos hay que mencionar la posible utilización en banca (selección de productos para clientes, análisis de riesgos y balances, reglas de auditoría e inspección, gestión de carteras, etc.), diseño (configuración de sistemas, diseño de circuitos etc.), diagnóstico de situación (diagnóstico médico, de averías de máquinas, del funcionamiento de una organización, etc.), enseñanza, entre otras aplicaciones.

6. Redes neuronales

En la segunda mitad de la década de los ochenta se produjo la reactivación de una tecnología de inteligencia artificial ya estudiada en los años cincuenta y sesenta que había estado prácticamente abandonada a finales de los años sesenta.

Como efecto de la atención creciente al proceso en paralelo y del reciente descubrimiento de algoritmos de aprendizaje nuevos para las simulaciones de sistemas de neuronas, se vuelve a confiar en uno de los primeros campos de estudio de la inteligencia artificial, al cual se hace referencia con la expresión *redes neuronales*.

Hoy día las **redes neuronales** constituyen una tecnología "renacida", todavía en desarrollo, en la cual se depositan grandes esperanzas a la hora de tratar con éxito algunos de los problemas clásicos de la inteligencia artificial, en particular el del reconocimiento de formas y de la palabra hablada.

6.1. Descripción

A continuación describiremos algunos aspectos más destacados de las redes neuronales.

Explicaremos cuáles son los modelos de redes neuronales, cuáles son sus características, qué aplicaciones tienen, así como algunas de sus particularidades.

6.1.1. Modelos de redes neuronales artificiales

Una **red neuronal** está constituida por un número variable de procesadores interconectados entre sí y que realizan una transferencia mutua de valores denominados *activaciones*.

Cada procesador recibe una serie de activaciones (**activaciones de entrada**) y, a partir de ellas, genera un valor de salida (**activación de salida**) que, a su vez, transfiere a otro grupo de procesadores que tiene conectados.

Por analogía con las redes biológicas, se da el nombre de sinapsis a las interconexiones y **neuronas** o **unidades** a los procesadores que forman los nodos de la red de procesadores, que recibe el nombre de **red neuronal**.

Esta descripción funcional se aplica tanto a las redes neuronales biológicas, como a su modelización eléctrica, es decir, a las **redes neuronales artificiales** (ANN, *artificial neural networks*) con las que se pretende reproducir, mediante sistemas electrónicos, la estructura y el comportamiento de las redes neuronales biológicas.

La costumbre ha llevado a omitir el calificativo "artificial" y, de manera general, se habla de redes neuronales para referirse a la modelización electrónica de las redes neuronales biológicas.

6.1.2. Algunas características de las redes neuronales

En el funcionamiento de las redes neuronales es posible distinguir la fase de ejecución y la fase de aprendizaje.

En la **fase de ejecución** cada neurona realiza una actualización de la activación de salida, de acuerdo con las activaciones de entrada que recibe en un momento dado. El funcionamiento durante esta fase se sintetiza en las ecuaciones que rigen la dinámica del sistema y el comportamiento de cada neurona en un momento dado.

Antes de usar o "ejecutar" la red neuronal, en la **fase de aprendizaje**, se cambian y actualizan los pesos que afectan las distintas activaciones de entrada con el objetivo de "enseñar" a la red neuronal a comportarse según se desea, es decir, a producir una salida determinada de acuerdo con la entrada recibida.

Es posible caracterizar las redes neuronales por:

a) La arquitectura de la red, determinada por la topología y la estructura que tiene y por el grado de conexión entre las neuronas que la forman.

b) La dinámica de la red, es decir, las ecuaciones que rigen el comportamiento de cada neurona, y en conjunto de la red, durante la fase de ejecución, es decir, cuando a la red ya se la ha "enseñado".

c) La regla, procedimiento o algoritmo de aprendizaje con el que se "enseña" a la red su cometido.

6.1.3. Otras denominaciones

Aunque el nombre más extendido es el de redes neuronales, los modelos como el indicado también han recibido otras denominaciones. A causa de la interconexión de varios procesadores simples se les ha denominado **modelos conexionistas** y, de hecho, representan el soporte de este planteamiento al campo de la inteligencia artificial.

A causa del paralelismo de la actuación conjunta de varios procesadores, también se habla de **modelos de proceso en paralelo distribuido**, como se hizo en un libro famoso de 1986, *Parallel Distributed Processing: Explorations in the microstructure of Cognition*, editado por **D. E. Rumelhart** y **J. L. McClelland**, que ha servido para la reactivación del interés sobre las redes neuronales.

Por la similitud estructural con el sistema nervioso biológico se los ha denominado también **sistemas neuromórficos** (*neuromorphic systems*) y se habla también de **cálculo neuronal** (*neural computing*) para referirse al aspecto de la informática y de la inteligencia artificial que trata de las redes neuronales y sus posibilidades.

6.1.4. Aplicaciones y funcionamiento

Las redes neuronales se utilizan principalmente en los sistemas clasificadores que han de dar respuesta a un estímulo concreto para reconocer su pertenencia o no a una clase determinada.

En concreto, parece que las redes neuronales son el mecanismo que muestra más posibilidades ante problemas como **el reconocimiento de formas** y **el reconocimiento de la palabra hablada** y, en general, ante todo tipo de problemas que requieran un alta capacidad de tratamiento efectuado en paralelo. En lugar de ejecutar una secuencia de instrucciones previamente almacenadas, como hacen los ordenadores que utilizan la arquitectura clásica debida a von Neumann, los modelos con redes neuronales exploran simultáneamente varias hipótesis utilizando sobradamente el proceso en paralelo, por lo cual también han servido para impulsar este aspecto de la arquitectura de los ordenadores.

> Se suele decir que la referencia a las redes neuronales se distingue de la informática clásica porque no se "programa" una red neuronal, sino que se "enseña" a una red neuronal a comportarse de una manera determinada.

La manera habitual de hacerlo es mostrar a la red neuronal varias señales de entrada junto con las salidas esperadas, y repetir el proceso más de una vez mientras la red neuronal "aprende" ajustando los pesos y las ponderaciones que determinan la dinámica de la red y el comportamiento de las distintas neuronas y sinapsis que la forman. El problema queda resuelto cuando se ha encontrado un conjunto de pesos que permiten a la red componer la salida adecuada para cada entrada, es decir, "reconocer" la entrada.

6.1.5. Neuroordenadores y *netware*

La modelización de una red neuronal puede tener lugar en el hardware utilizando circuitos de proceso en paralelo y, a tal efecto, se utilizan arquitecturas especiales para el proceso en paralelo como, por ejemplo, los ordenadores matriciales. También se diseñan chips VLSI que incorporan la estructura de una red neuronal. Generalmente se asigna el nombre de neuroordenadores (*neurocomputers*) a estos sistemas, y se suelen utilizar como coprocesadores especializados y auxiliares de los ordenadores digitales tradicionales. En este caso se accede a la red neuronal como si fuera una subrutina del sistema.

En el caso de los *neurocomputers*, no tiene sentido hablar de su potencia haciendo referencia a los millones de instrucciones por segundo (mips) que son característicos de los ordenadores con arquitectura von Neumann. En los neuroordenadores se habla de **interconexiones por segundo** y, por ejemplo, el sistema Neuro-07 anunciado por NEC en 1988 ya incorporaba un neuromotor (*neuro-engine*) de 210.000 interconexiones por segundo. El sistema incluía un ordenador personal convencional con pantalla de color, un neuromotor con su teclado particular y software para "enseñar" a la red neuronal, todo por un precio de poco más de un millón de pesetas (unos 6.000 euros).

El *netware* es un software que simula las neuronas y su interconexión en un ordenador convencional con arquitectura von Neumann.

En el mismo año al que hacíamos referencia, 1988, se vendieron unos 10.000 paquetes de *netware* neuronal en Estados Unidos.

La década de los años ochenta

A finales de la década de los años ochenta, datos publicados en agosto de 1989 estimaban ya en unas 300 las empresas norteamericanas especializadas en la tecnología de las redes neuronales, y varias de ellas se dedican, precisamente, a elaborar *netware* para ordenadores personales y estaciones de trabajo.

Incluso con estaciones de trabajo, y también con los ordenadores personales de la microinformática, es posible la investigación y utilización de pequeñas redes neuronales (menos de cincuenta elementos procesadores o neuronas en un PC compatible, según algunos autores).

Aunque se ha de tener en cuenta las limitaciones de proceso de la arquitectura von Neumann sometida a las exigencias de simular el proceso en paralelo de las redes neuronales, y que es importante disponer de compiladores adecuados para la simulación del proceso en paralelo, los PC resultan ser adecuado para el estudio y la investigación con redes de dimensiones reducidas.

6.1.6. Características particulares de las redes neuronales

Gracias al elevado grado de paralelismo, las redes neuronales presentan una tolerancia a errores elevada (*fault tolerance*) a causa de la redundancia de elementos procesadores y a las características de su dinámica de funcionamiento. Por ello también presentan un proceso de degradación controlado y limitado (*graceful degradation*) cuando se presenta el error de uno de los procesadores.

Además, cabe recordar que las redes neuronales no son eficientes para realizar las tareas habitualmente encomendadas a los ordenadores de proceso secuencial (cálculos con gran exactitud, manipulación de cifras y datos, etc.), pero resultan mejores para el manejo de conjuntos incompletos de datos o de información difusa o contradictoria.

6.2. Arquitectura y topología de las redes neuronales

A continuación explicaremos cuál es la estructura topológica y la arquitectura en las redes neuronales.

6.2.1. Estructura topológica

Con respecto a la arquitectura y topología de la red, se puede hablar, por ejemplo, de redes con alimentación hacia delante (*feed forward*) o de redes con retroalimentación (*feedback*). En las primeras, el sentido de propagación de las activaciones en la fase de ejecución es único (hacia delante), mientras que en las segundas, las activaciones de salida de unas neuronas se utilizan también como activaciones de entrada en un circuito de retroalimentación (podéis ver la figura 9 al margen).

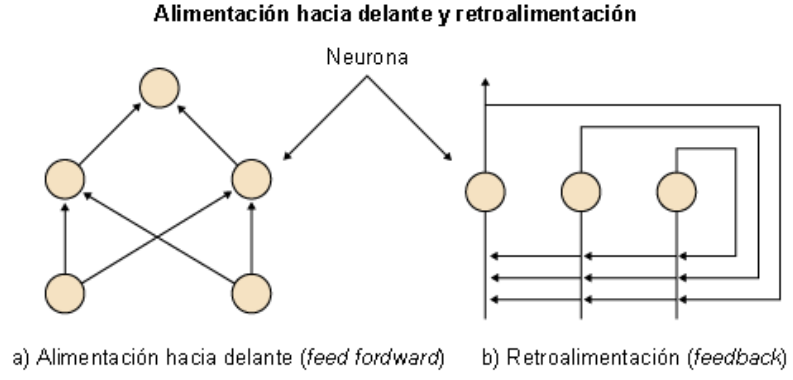


Figura 9

También se pueden diseñar redes en las cuales se forman estructuras jerárquicas entre las neuronas o utilizar cualquier otra topología.

Un caso particular de estas redes es el que se ha denominado redes de estructuras competitivas, en el cual las neuronas se agrupan en *clusters* y dentro de cada uno de ellos, tan sólo una neurona puede estar activa en un momento determinado gracias al carácter inhibidor de las sinapsis entre ellas (podéis ver la figura 10 al margen).

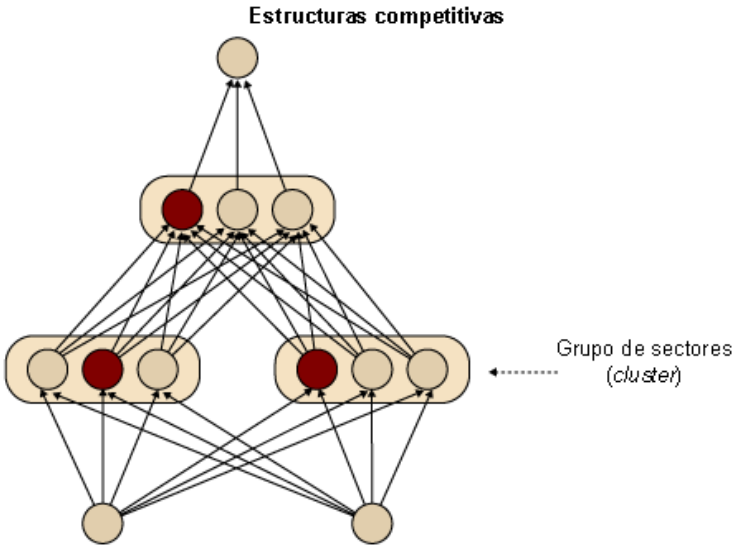


Figura 10

6.2.2. Estructura en capas de una red neuronal

En el caso más utilizado, las neuronas en una red neuronal suelen estructurarse de manera que integren una o varias "capas" o niveles (*layers*). Según los diseños, hay redes de una, dos y varias capas.

La primera capa se denomina **capa de entrada** (*input layer*) y es la que recibe los estímulos del exterior. Si hay más capas, la última de ellas es la **capa de salida** (*output layer*), de la cual se obtiene el resultado de la fase de ejecución de la red neuronal. Si hay más de dos capas, entre la primera y la última hay un conjunto de capas ocultas (*hidden layers*) formadas por neuronas que almacenan la información en una "representación interna", y por ello también reciben el nombre de unidades de representación interna (*internal representation units*) (podéis ver la figura 11 al margen).

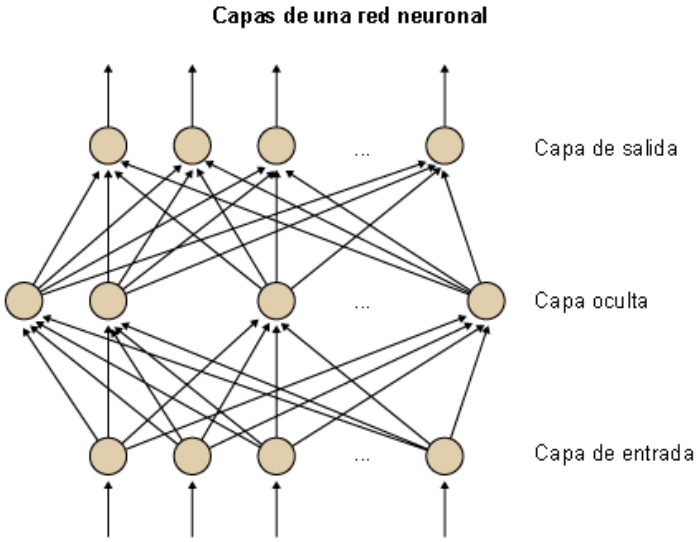


Figura 11

Las capas pueden estar constituidas por un número cualquiera de neuronas, normalmente impuesto por las características del problema que se ha de resolver o por las prestaciones que se desee obtener de la red neuronal. Así pues, en el diseño de una red neuronal se dan dos primeros grados de libertad al elegir tanto el número de capas, como el número de neuronas en cada capa.

Las dos espirales

Un ejemplo posible es el que se muestra en la figura 12, con dos neuronas de entrada, una de salida y dos capas ocultas de cinco neuronas cada una. K. Lang y M. Witbrock, de la Carnegie Mellon University, utilizaron esta red para resolver el problema conocido como "de las dos-espirales", que consiste en reconocer si un punto pertenece o no a dos espirales entrelazados. Gracias a la simplicidad del modelo, ha sido posible estudiar el comportamiento de las capas ocultas en una red neuronal y profundizar en el conocimiento de la representación interna de la información en una red con más de dos capas, uno de los problemas más interesantes y sugestivos en la investigación sobre redes neuronales.

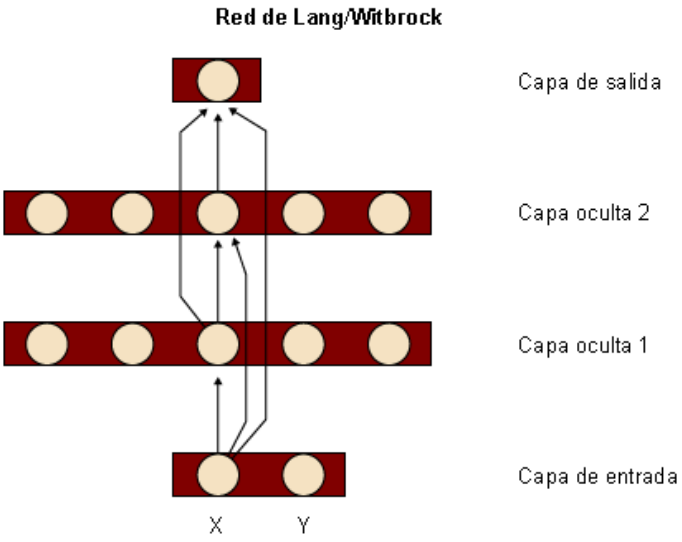


Figura 12

6.2.3. Conexiones entre las neuronas

Las sinapsis o conexiones entre las neuronas que forman las distintas capas de la red se pueden realizar de varias maneras, lo cual influye en la topología y arquitectura de la red y en su funcionamiento.

El sistema más habitual sigue el principio de la conectividad total, según el cual cada neurona perteneciente a una cierta capa está conectada con todas y cada una de las neuronas que forman la capa inmediatamente inferior y de todas ellas recibe activaciones de entrada.

Además, cada neurona de una capa está conectada también con todas y cada una de las neuronas que integran la capa inmediatamente superior, a las cuales transfiere su activación de salida. Así se mostraba al conexionado de la figura 11 que repetimos al margen, con una única capa oculta.

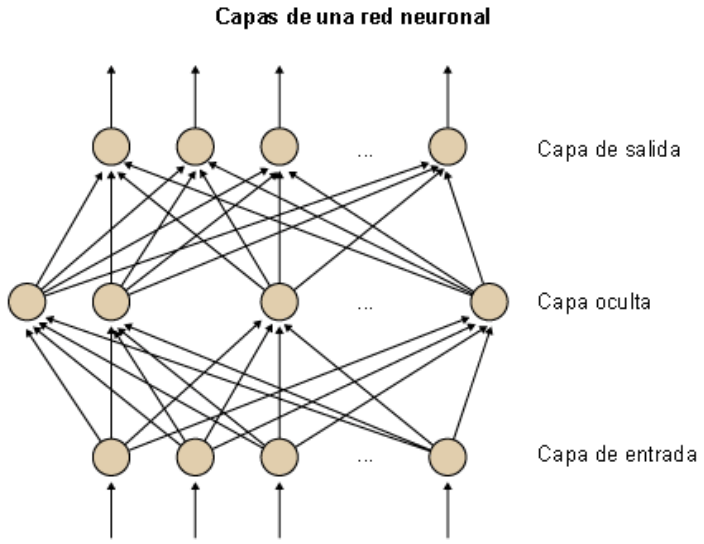


Figura 11

6.3. Dinámica y aprendizaje de una red neuronal

A continuación explicaremos la dinámica y el aprendizaje de una red neuronal en dos fases: fase de ejecución y fase de aprendizaje.

6.3.1. Fase de ejecución

En la fase de ejecución cada interconexión o sinapsis tiene un peso establecido obtenido en la fase de aprendizaje.

La dinámica de funcionamiento de la red neuronal establece que la señal de entrada en cada neurona es afectada por el peso mencionado en una combinación lineal de todas las activaciones de entrada que recibe la neurona. Una

vez efectuada esta combinación lineal, se añade al resultado una cierta función umbral característica de la neurona. Al resultado obtenido se aplica una función propia de la neurona que es la que determina finalmente la activación de salida. En el caso, hoy día frecuente, de varias capas de neuronas conectadas siguiendo el principio de la conectividad total, la fórmula que determina el comportamiento dinámico de la red neuronal es:

$$a_k(c) = f(\Sigma [W_{ki}(c) * o_k(c - 1)] + u_k(c))$$

donde:

ak(c) es la activación de salida de la neurona de la columna **k** a la capa **c**;

W_{ki}(c) es el peso asignado a la interconexión entre la neurona de la columna **k** de la capa **c** y la neurona de la columna **i** de la capa **c-1**;

o_k(c-1) es la activación de entrada para esta interconexión y es igual a **i_k** (la señal de entrada en la columna **k** de la capa de entrada) cuando la capa es la primera (c = 0), e igual a **a_k (c-1)** en las otras capas;

u_k(c) es la función umbral para la neurona de la columna **k** de la capa **c**; y finalmente,

f() es la función de activación de la neurona.

Todo esto permite esquematizar gráficamente una neurona con sus activaciones en el diagrama de la figura 13 en el margen.

Activaciones y comportamiento dinámico

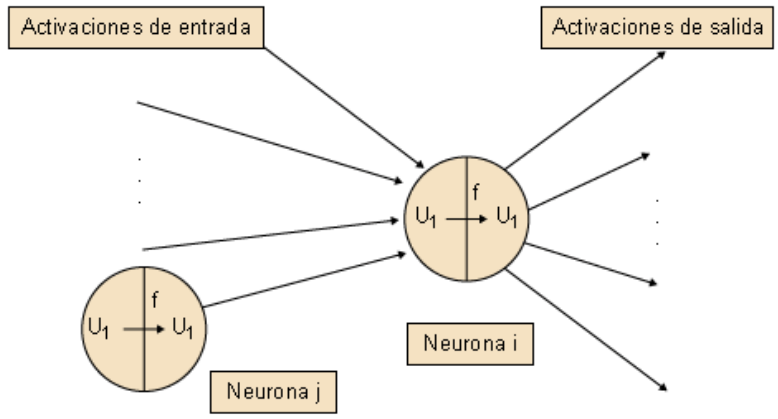


Figura 13

La función **f()**, que define el comportamiento de una neurona en una red neuronal, puede ser **determinista** o **estocástica** según si el potencial postsináptico o la activación de salida se obtenga de manera determinada de la entrada o de acuerdo con una distribución probabilística.

Las funciones deterministas pueden ser lineales como f(x) = x, pero en la mayoría de los casos son no lineales. Las funciones más utilizadas son:

a) Funciones de tipo binario si la respuesta es 0 ó 1.

b) Funciones isign o *hard delimiter* si la respuesta es +1 o –1, es decir, del tipo f(x) + sign(x).

c) Funciones sigmoides, generalmente según la tangente hiperbólica en la forma f(x) = th(x).

La fase de ejecución puede ser paralelizada fácilmente (incluso en una arquitectura von Neumann) mediante una partición del proceso de cálculo. Esto se consigue utilizando un procesador (o su simulación) para cada neurona, y el procesador mencionado se encarga de almacenar en una memoria local los pesos de ponderación asociados a la neurona que debe simular. Para ello, cada procesador toma la activación de la capa inmediatamente inferior y la multiplica por el peso correspondiente. La activación obtenida se transfiere al procesador siguiente mientras se recibe la nueva activación procedente del procesador anterior que, a su vez, se multiplica (***** en la fórmula) por el peso correspondiente y se acumula (S en la fórmula) al resultado anterior. Este proceso de multiplicación y acumulación se repite hasta que se obtienen todos los productos parciales indicados en la ecuación.

6.3.2. Fase de aprendizaje

En la fase de aprendizaje se "enseña" a la red neuronal el comportamiento deseado.

Las reglas de aprendizaje pueden ser supervisadas cuando existe un "profesor" que dirige el entrenamiento de la red e indica si la respuesta al estímulo de entrada es correcta o no; o autoorganizadas cuando no hay un "profesor" externo a la red. En este último caso se habla, más frecuentemente, de una regla de aprendizaje no "supervisada".

La regla de aprendizaje

En algunos casos se prescinde del proceso de aprendizaje y se dice que la regla de aprendizaje es "instantánea". Pero, en la mayoría de los casos, la regla o el algoritmo de aprendizaje es de gran importancia.

Hay varios algoritmos o reglas para proceder a la actualización de los valores de los pesos de las interconexiones o sinapsis durante el proceso de entrenamiento de la red. Cada algoritmo da lugar a una cierta estructura y comportamiento de la red neuronal.

El problema central en la eficacia de un algoritmo de aprendizaje es determinar cuáles son las interconexiones que tienen una modificación de pesos que supone una mejora rápida de la eficiencia de la red neuronal. El problema resulta especialmente complejo cuando se trata de interconexiones en redes neuronales de varias capas y hay "capas ocultas" que mantienen una "representación interna" desconocida de la información que circula por la red neuronal. Precisamente, la difusión a finales de los años ochenta de uno de estos algoritmos de aprendizaje se considera una de las causas principales del renacimiento del interés por las redes neuronales.

El mencionado algoritmo es conocido como el de la propagación de errores hacia atrás (*back propagation of errors*), pero suele abreviarse en la literatura especializada como *back propagation* e, incluso, como *back-prop*. Se trata de un algoritmo supervisado que propuso **D. E. Rumelhart** y se puede aplicar a redes de varias capas, aunque habitualmente se utilizan redes de tres capas en la mayoría de los estudios y las investigaciones. Este algoritmo presenta, entre otras, la ventaja de permitir un amplio rango de aplicaciones, principalmente a causa del hecho de que también acepta entradas y salidas continuas.

En primer lugar se establecen al azar los valores de los pesos y se ofrece una entrada a la red neuronal que producirá actividad en cada una de las capas de la red. Un "profesor" que conoce cuál ha de ser la respuesta de cada neurona de la capa de salida indica en cada una de ellas la amplitud y el signo de su error. Las mencionadas señales de error se utilizan para ajustar los pesos de todas las interconexiones entre la capa de salida y la capa oculta inmediatamente anterior a ella, de manera que el error obtenido en la capa de salida se propaga hacia atrás, hacia las capas ocultas que, a su vez, utilizan los valores de error recibidos para ajustar el valor de sus sinapsis. Así pues, en la fase de aprendizaje el flujo de datos tiene un sentido contrario al de la fase de ejecución y se propaga "hacia atrás".

De hecho, el algoritmo consigue realizar pequeños ajustes en cada sinapsis de la red, de manera que el conjunto de las modificaciones reduce el error total en el funcionamiento de la red. Una vez se ha aplicado el mismo procedimiento varias veces, se llega a un valor mínimo del error total. Como todos los métodos de gradiente descendente, se podría tratar de un mínimo local, pero eso raramente ocurre en las redes neuronales, posiblemente por el número de neuronas involucradas y la gradualidad de su respuesta.

6.4. Historia de las redes neuronales

A continuación haremos una breve introducción a la historia de las redes neuronales, desde los primeros estudios que se hicieron sobre el tema hasta el nuevo interés que ha despertado en nuestros tiempos.

6.4.1. El *perceptron* de F. Rosenblatt

Los primeros estudios sobre redes neuronales datan de 1943, cuando **McCulloch** y **Pitts** iniciaron la interpretación mecanicista de las redes neuronales. Pero sólo a partir de los años sesenta las redes neuronales empezaron a ser consideradas como una posible solución a los problemas de realizar clasificaciones automatizadas. Para ello fue decisivo el trabajo de **Franz Rosenblatt**, un psicólogo de la Universidad Cornell que, por medio del análisis matemático, la simulación en ordenadores digitales y algunos experimentos con sistemas analógicos de proceso en paralelo, llegó a mostrar cómo se podía enseñar a actuar como clasificadores a las redes neuronales con valores variables en los pesos de las interconexiones.

El ***perceptron***, desarrollado por Rosenblatt a partir de 1958, era en realidad un clasificador de categorías aplicado a la simulación de la visión humana. Rosenblatt proponía una regla de aprendizaje que, aunque era lenta, tenía que convergir siempre que existiera una solución. En realidad, el modelo de *perceptron* con dos capas de neuronas podía computar, por ejemplo, las funciones lógicas *or* y *and*, pero no la función *xor*. La introducción de una tercera capa, una capa oculta, aportaría la potencia suficiente para implementar esta función. Pero un artículo de **Minsky** y **Papert** de 1967 demostraba que el procedimiento de convergencia de Rosenblatt sólo podía resolver problemas linealmente separables, lo que provocó que en la práctica se abandonara durante varios años esta vía de investigación.

6.4.2. El predictor meteorológico de Widrow

A pesar del descrédito que el trabajo de **Minsky** y **Papert** aportó a la teoría de los perceptrones, otro de los pioneros que trabajaba desde los años cincuenta, **Bernard Widrow**, de la Universidad de Stanford, implementó en 1963 una red neuronal que actuaba como un **sencillo sistema de predicción meteorológico** capaz de dar predicciones correctas en un 85% de los casos, frente al estimado 65% que obtenía el hombre del tiempo local. Widrow utilizaba una regla de aprendizaje con una función de "coste" que debía ser minimizada. La función de coste mencionada medía la diferencia entre la salida real de la red neuronal en su capa de salida y el valor esperado.

Pero la realidad es que, después del influyente artículo de Minsky y Papert, los perceptrones y las redes neuronales fueron prácticamente abandonadas durante los años setenta, e incluso se llegó a dudar de la posibilidad de continuar considerándolos como objetos válidos de investigación.

6.4.3. El nuevo interés por las redes neuronales

Se puede decir que el interés por las redes neuronales renació en el año 1982 a partir del éxito de los trabajos de **J. J. Hopfield** sobre redes neuronales. El modelo de Hopfield introdujo las redes neuronales que seguían el principio

de la conectividad total y utilizaba señales binarias y con una arquitectura realimentada. La red neuronal de Hopfield, en sus varias versiones, se puede utilizar como una forma de memoria asociativa y también como un sistema para resolver problemas de optimización.

Otro sistema con gran éxito e influencia fue la red **NETTalk**, presentada en 1986 por T. J. Sejnowski y C. R. Rosenberg, de la Universidad John Hopkins. Se trata de una red neuronal de tres capas que tiene como objetivo aprender a pronunciar el inglés. La entrada es texto que ha de pronunciar una máquina que puede producir sonidos. Aunque al principio la red balbucea y se equivoca, gracias al entrenamiento y la corrección de los pesos de las interconexiones, acaba pronunciando correctamente, incluso al intentarlo con textos con los que no ha sido entrenada (sólo registra un 10% de errores, posiblemente porque la pronunciación inglés a veces depende del contexto).

Con este sistema también se demostró que el número de neuronas en capas ocultas es de gran importancia: pocas neuronas impiden que la red neuronal cumpla su función, pero con un exceso de neuronas en la capa oculta la red puede funcionar un poco mejor, aunque no sabe generalizar (en el caso de la NETTalk, es incapaz, por ejemplo, de leer textos que no haya preparado previamente).

También se considera de gran influencia la difusión en 1986 del **algoritmo *back propagation*** y el trabajo recapitulativo sobre el proceso paralelo distribuido editado por Rumelhart y McCleland. Nuevos investigadores como Feldman (propiedades de los modelos conexionistas), Grossberg (el cerebro adaptativo) y muchos otros han proporcionado un gran impulso al estudio de las redes neuronales.

Los hechos que resumen el nuevo interés se centran en el desarrollo de nuevas topologías y algoritmos, la posibilidad de implementaciones en circuitos VLSI, los éxitos de sistemas como el NETtalk de Sejnowski y Rosenberg y de las mismas redes de Hopfield y, también, la creciente fascinación sobre el funcionamiento del cerebro humano que las redes neuronales ayudan a estudiar. En cualquier caso, la redes neuronales son, otra vez, una tecnología en crecimiento de la que todavía resulta prematuro hacer hipótesis sobre su futuro.

6.5. Las redes neuronales y el cerebro humano

La similitud estructural entre las redes neuronales biológicas y las redes neuronales artificiales anima a utilizar estas últimas para entender mejor el cerebro humano y la dinámica de su funcionamiento.

En cualquier caso, la analogía no puede ser completa y hay que tener en cuenta muchos factores de diferenciación como, por ejemplo, las evidentes limitaciones de tamaño de las redes artificiales. En un cerebro humano se estima que hay en torno a 10^{11} neuronas que presentan una conectividad media del

orden de 1.000; esto sitúa el número de interconexiones o sinapsis en 10^{14}, cifras por ahora inalcanzables con los modelos artificiales que se ven reducidos a casos exageradamente simples.

Por otra parte, el algoritmo de propagación hacia atrás (*back propagation*), que tanto éxito ha tenido en las redes neuronales artificiales, no parece realista en el ámbito del cerebro humano, en el que la activación de salida de una neurona (al menos en el neocórtex) es excitadora o inhibidora, pero no las dos al mismo tiempo.

Otro problema importante que surge al asimilar el comportamiento de las redes neuronales artificiales con el del cerebro humano es la exigencia de una parte diferenciada del cerebro humano, la que haría de "profesor" en el proceso de aprendizaje, ya que ha de tratarse de neuronas con propiedades nuevas y específicas que parece que no existen en el cerebro humano.

Estas razones y muchas otras que no hace falta mencionar aquí hacen pensar a algunos especialistas como **F. Cricks** que estos supuestos "modelos" neuronales no modelizan nada real en el cerebro humano, y que otra vez se trata de sistemas que "manifiestan" en algunos casos un comportamiento similar al cerebro humano, sin que de ello se deduzca que se trate de un modelo real.

De hecho, aunque las redes neuronales biológicas no utilicen, por ejemplo, un algoritmo como el de propagación hacia atrás, lo cierto es que el comportamiento final es similar en algunos casos. No hay que olvidar que las redes neuronales resultan el sistema artificial más eficiente para el reconocimiento de formas y, curiosamente, ésta es una actividad cerebral muy sencilla que incluso los animales son capaces de realizar fácilmente, aunque, por la gran potencia de cálculo que exige, ha fracasado el intento de resolverla con sistemas artificiales (a menudo basados en la abstracción matemática) que no sean las redes neuronales.

7. Lenguajes específicos para la IA

A continuación damos constancia de los lenguajes específicos para la inteligencia artificial más relevantes.

7.1. Lisp

Es frecuente la consideración de que el Lisp es el lenguaje propio de la inteligencia artificial a causa de sus características particulares, su existencia desde 1962 y por haber estado sobradamente utilizado por los investigadores en el campo de la IA.

7.1.1. Origen

El Lisp se desarrolló en el MIT a raíz de un trabajo publicado por John McCarthy en 1960: *Recursive functions of symbolic expresions and their computation by machine*. Posteriormente, en 1962, se publicó el primer manual del Lisp para el equipo IBM 7090 y sus autores fueron el mismo **McCarthy junto con Abrahams, Edwards, Hart y Levin**, todos ellos del MIT. El lenguaje utiliza también las llamadas funciones "lambda" surgidas del *lambda calculus*, un lenguaje formal sobre funciones desarrollado por **A.Church**.

7.1.2. Objetivos y características principales

El Lisp implementa el tratamiento mediante expresiones simbólicas y recursividad y dispone de una gran potencia y simplicidad en la definición de estructuras. En Lisp no hay diferencias entre las estructuras de datos y las estructuras de programa; las dos se reducen a **expresiones simbólicas** que tienen como casos particulares los **átomos** y las **listas**. De aquí el nombre de Lisp derivado de ***list p**rocessor* (procesador de listas).

En realidad, el Lisp se puede tratar como un lenguaje matemático formal que, con las extensiones adecuadas, se convierte en un lenguaje de programación. Quizá por ello es un lenguaje sumamente flexible, e incluso lo puede cambiar el mismo programador.

Se trata, en sus inicios, de un **lenguaje funcional** que no tiene por qué disponer de la instrucción de asignación, cuyos programas son funciones que se definen por composición de otras funciones más simples. Para ejecutarlo, se "aplica" a los datos de entrada, que son los parámetros de la función, para obtener un resultado, que es el valor calculado de la función.

Las variantes del Lisp más utilizadas en la práctica también incorporan muchos elementos no funcionales en nombre de la eficiencia.

7.1.3. Dialectos

Un problema habitual para los usuarios del Lisp es que no existe ninguna estandarización válida y hay varios dialectos en uso. Se reconoce un núcleo central, denominado en ocasiones Pure Lisp, y variantes como el Maclisp, utilizado en el MIT; el PSL, con pretensión de estandarización portable desarrollado en la Universidad de Utah; el Le-Lisp, versión utilizada en Francia, etc.

Los intentos de estandarización se iniciaron sin mucho éxito con el Standard Lisp de Hearn, en 1966, y llegaron a un consenso con el Common Lisp aparecido en 1984. También cabe destacar los entornos de programación Lisp, de entre los cuales el Maclisp es el que se convirtió en uno de los más conocidos al ampliarse con variantes como el Franzlisp o el Zetalisp. También es posible mencionar el Interlisp, de la Universidad de Cambridge.

El lenguaje **Logo** es, en el fondo, una derivación simplificada del Lisp para su utilización en la enseñanza, donde se utiliza para crear los **entornos de descubrimiento**.

7.2. Prolog

El Prolog es otro de los lenguajes específicos más relevantes que hay sobre inteligencia artificial.

7.2.1. Orígenes

A comienzos de los años setenta, los partidarios de la programación lógica se dedicaron a implementar en el ordenador un sistema que gestionara la lógica causal que es, en realidad, una restricción de la lógica de predicados de primer orden (cláusulas de Horn). El lenguaje que implementaba la mencionada **PROgramación LÓGica** se denominó Prolog y nació en la Universidad de Marsella, bajo la dirección de **Alan Colmerauer**.

Algunos hitos fundamentales en la historia del Prolog son un primer programa para la prueba de teoremas escrito en 1973 por Colmerauer en Fortran; la formulación completa de la programación lógica de **Robert A. Kowalski** en 1974; la implementación de un Prolog eficiente en un PDP-10 realizada en 1977 por **David H. D. Warren** y, finalmente, la estandarización de la llamada **sintaxis de Edimburgo** en el libro *Programming in Prolog*, de **Cloksin** y **Mellish** en 1981.

Un elemento adicional a favor de la difusión y la fama del Prolog fue su uso anunciado en el proyecto japonés de la "quinta generación" de ordenadores.

7.2.2. Objetivos y características principales

La implementación de la lógica causal se consigue con una representación del conocimiento a base de hechos y reglas con la forma:

HECHOS: P(X1..., Xn)
REGLAS: P(X1..., Xn):- P1(...)..., Pm(...)
(donde ":-" se lee como "**si**")

que constituyen los elementos almacenados en la base de datos (conocimientos en forma de reglas y hechos en forma de hechos) del sistema Prolog.

Por lo tanto, se trata de un sistema de producción basado en reglas y es un **lenguaje declarativo** en el cual se indican los hechos y las reglas y no el procedimiento (algoritmo) con el que se encuentran las soluciones.

El usuario proporciona los hechos y las reglas y el Prolog incorpora un motor de inferencia que implementa los mecanismos de recursividad, unificación y razonamiento por encadenamiento hacia atrás (*backtracking*). Eso se hace patente en el modo de utilizar el Prolog una vez implementada la base de hechos y reglas: preguntar si es cierto o no un objetivo determinado (*goal directed*). Todo esto provoca que el Prolog sea especialmente adecuado para la implementación de sistemas expertos basados en reglas, y en la que sea correcto el uso de este mecanismo de razonamiento.

7.2.3. Dialectos

La problemática de los muchos dialectos del Lisp no aparece en el Prolog, en cuyo uso es frecuente utilizar la sintaxis de Edimburgo, establecida en 1981 en el libro ya mencionado de Cloksin y Mellish.

7.3. Smalltalk

Aunque el lenguaje Smalltalk se creó inicialmente para otros objetivos, también se ha considerado con gran interés en el ámbito de la IA. Se trata de un lenguaje concebido como soporte de la programación orientada a objetos y que presenta la ventaja de cumplir los requisitos generales de un lenguaje para IA: incorpora tratamiento simbólico y dispone de una estructura de *array* muy similar a las listas del Lisp.

Gracias a su adaptación a la programación orientada a objetos, el Smalltalk dispone de módulos que se transmiten **mensajes** que vienen a ser órdenes de ejecución de operaciones. Cada **objeto** se define mediante el conjunto de **variables** que constituyen su estado interno y de los **métodos** asociados a las operaciones que servirán para manipular las variables mencionadas.

También es de gran importancia la facilidad del Smalltalk para utilizar el concepto de **herencia**. La definición de un objeto en el Smalltalk implica indicar su clase, ya que todo objeto es una *instancia* de una clase. Las clases se organizan jerárquicamente y toda subclase "hereda" las variables y los métodos de la clase superior.

Algunos autores apuestan por el futuro de un sistema denominado **Loops**, que integra el uso del Smalltalk con el Lisp y el Prolog, aunque cualquier decisión es prematura en este sentido.

Actividades

Una actividad complementaria al alcance de muchos es atreverse a jugar al ajedrez contra un programa informático. Sin necesidad de llegar a la sofisticación del *Deep Blue*, el programa al cual se ha enfrentado varias veces el campeón humano Gari Kasparov, la realidad es que los jugadores no profesionales suelen perder ante programas de ajedrez incluso sencillos. Y éste no es más que un primer ejemplo de las técnicas de la inteligencia artificial en funcionamiento.

Para llegar a hacerse una idea de cómo podría ser un mundo con programas y máquinas inteligentes, es recomendable la lectura de la buena narrativa de ciencia-ficción en torno a los robots.

La mejor opción, evidentemente, son los relatos y las novelas de robots del científico, divulgador y novelista norteamericano Isaac Asimov, creador de las *Tres leyes de la robótica*, verdadero código ético para los robots inteligentes del futuro. Una propuesta de lectura divertida y sugerente, que recomendamos especialmente, sería:

Relatos:

Asimov, Isaac (1950). *Yo robot*. Barcelona: Plaza y Janés.

Asimov, Isaac (1982). *Los robots*. Barcelona: Martínez Roca.

Novelas:

Asimov, Isaac (1953). *Bóvedas de acero*. Barcelona: Plaza y Janés.

Asimov, Isaac (1957). *El sol desnudo*. Barcelona: Plaza y Janés.

Otra opción, menos optimista, es la lectura de algunas críticas muy serias que han surgido en contra del proyecto de la llamada "inteligencia artificial dura". En resumen, estas críticas plantean la imposibilidad del proyecto de la inteligencia artificial.

En este sentido, resulta recomendable la lectura, no tan sencilla pero muy interesante y estimulante, del libro:

Penrose, Roger (1989). *La nueva mente del emperador*. Barcelona: Mondadori, 1991.

Ejercicios de autoevaluación

1. ¿Cuál es el objetivo principal de la inteligencia artificial?
2. Ya se han construido programas que superan el test de Turing. ¿Verdadero o falso?
3. Los programas de inteligencia artificial son capaces de aprender. ¿Verdadero o falso?
4. Mencionad un ejemplo de programas de inteligencia artificial al alcance del gran público.
5. La inducción es la "inferencia lógicamente correcta". ¿Verdadero o falso?
6. ¿Qué es la heurística?
7. ¿Qué es la base de conocimientos de un sistema experto?
8. Los ingenieros del conocimiento son los expertos cuya experiencia emulan los sistemas expertos. ¿Verdadero o falso?
9. Mencionad dos lenguajes específicos de la inteligencia artificial.
10. Una red neuronal artificial suele tener miles de neuronas. ¿Verdadero o falso?

Solucionario

Ejercicios de autoevaluación

1. El objetivo principal de la inteligencia artificial es obtener sistemas informáticos con un comportamiento que consideraríamos inteligente en un ser humano.
2. Falso. A pesar de las expectativas del mismo Turing, que creía que más o menos hacia el año 2000 se superaría el test que lleva su nombre.
3. Verdadero. El aprendizaje es uno de los objetivos importantes de la inteligencia artificial.
4. Algunos programas informáticos que juegan al ajedrez utilizan técnicas de inteligencia artificial.
5. Falso. La "inferencia lógicamente correcta" es la deducción (*modus ponens* o instanciación universal).
6. La heurística es el conjunto de criterios, métodos o principios que se utilizan para encontrar, entre varios caminos posibles, cuál o cuáles son los más efectivos para obtener un objetivo determinado. Viene a significar las "reglas de la experiencia" para seleccionar un camino de acción.
7. La base de conocimientos de un sistema experto es el conjunto de reglas que sintetizan la experiencia del experto humano que se quiere emular.
8. Falso. Son los especialistas en inteligencia artificial que ayudan a transferir la experiencia y el conocimiento de un experto humano a un sistema experto.
9. Lisp y Prolog.
10. Falso. Con muy pocas neuronas se obtienen buenos resultados. La red de Lang/Witbrock, por ejemplo, utiliza sólo trece neuronas.

Bibliografía

Bibliografía básica

Cloksin, W. F.; Mellish, C. S. (1981). *Programming in PROLOG*. Nueva York: Springer-Verlag.

Cortés, U. [*et al*.] (1995). *Inteligencia artificial*. Barcelona: Ediciones UPC (Politexto, 17).

Cortés, U.; Sierra, C. (1987). *LISP*. Barcelona: Marcombo-Boixareu Editores.

Jackson, P. (1986) *Introduction to Expert Systems*. Reading, Massachusetts: Addison-Wesley.

McCorduck, P. (1979). *Máquinas que piensan: una incursión personal en la historia y perspectivas de la inteligencia artificial*. Madrid: Tecnos, 1991.

Wasserman, P. (1989). *Neural Computing, Theory and Practice*. Nueva York: Van Nostrand Reinhold.

Bibliografía complementaria

Charniak, E.; McDermott, D. (1992). *Introduction to Artificial Intelligence* (2.ª ed.). Reading, Massachusetts: Addison-Wesley.

Feigenbaum, E.; McCorduck, P. (1983). *The Fifth Generation*. Reading, Massachusetts: Addison-Wesley.

Goldberg, A.; Robson, D.; Ingalls, D. H. (1984). *SMALLTALK 80: The Languaje and Its Implementation*. Reading, Massachusetts: Addison-Wesley.

Hancock, P. A.; Chignell M. H. (eds.) (1989). *Intelligent Interfaces: Theory, Research, and Design*. Amsterdam: North-Holland.

Hofstader, D. R.; Dennett, D. C. (eds.) (1985). *The Mind's I*. Harmondsworth, Middx: Basic Books Inc., Penguin Books Ltd.

Kowalski, R. A. (1979). *Logic for Problem Solving*. Amsterdam: North-Holland.

Newell. A; Simon, H. (1972). *Human Problem Solving*. Englewood Cliffs, N. J.: Prentice-Hall.

Rich, E. (1992). *Artificial Intelligence*. Singapore: McGraw-Hill, International Student Editon.

Siklóssy, L. (1976). *Let's Talk LISP*. Englewood Cliffs, N. J.: Prentice-Hall.

Wertz, H. (1986). *LISP: Introducción a la programación*. Barcelona: Masson.

Casos prácticos sistemas informáticos

Lluís Anaya Torres

PID_00153117

Universitat Oberta de Catalunya
www.uoc.edu

Índice

Objetivos

1. Conocer las diferencias entre una empresa orientada al servicio y una empresa orientada al productor.

2. Distinguir entre los sistemas informáticos comunes y los específicos de las empresas orientadas a los servicios y al producto.

3. Aplicar los conceptos teóricos aprendidos en módulos anteriores, considerando que los casos de este módulo se pueden utilizar en el estudio de la asignatura según las indicaciones de los consultores en cada semestre.

1. Empresas de productos y empresas de servicios

1.1. Organización y sistemas informáticos

Una definición simple, breve y clara de las organizaciones podría ser la siguiente: "Las organizaciones son personas (recursos humanos) que hacen cosas (procesos) para personas (clientes)".

Con esta simple definición, podemos ubicar los sistemas informáticos como la herramienta que utilizan las personas de la organización (los recursos humanos) para hacer cosas (procesos). Concretamente, los sistemas informáticos son una herramienta potente que ayuda y da apoyo al modelo de gestión de las organizaciones. La misión de los sistemas informáticos dentro de una organización debe ser la de contribuir a mejorar la eficiencia de los procesos de las organizaciones y a ofrecer información útil para la toma de decisiones.

Cuando queramos informatizar una parte de una empresa, tenemos que alcanzar la misión, porque en caso contrario no tiene sentido la inversión. Para autoevaluar la conveniencia de una informatización empresarial, siempre tendremos que conseguir una respuesta afirmativa a la hora de preguntarnos lo siguiente:

- ¿Mejoraremos la eficiencia de un proceso con la informatización que queremos realizar?

- ¿Obtendremos información útil para la toma de decisiones con la informatización que queremos realizar?

1.2. Tipos de organizaciones. Tipos de sistemas informáticos

Existen muchos tipos de organizaciones diferentes y podemos establecer muchas clasificaciones. Sin embargo, siguiendo con la visión simple, breve y clara con la que hemos empezado, podríamos clasificar a las organizaciones en dos categorías, dependiendo de lo que vendan:

- **Empresas de servicios**. Son aquellas organizaciones que venden a clientes bienes inmateriales o intangibles, como por ejemplo una gestoría administrativa, una empresa consultora informática, un despacho de abogados o un ayuntamiento.

- **Empresas de productos**. Son aquellas organizaciones que venden a clientes bienes materiales o tangibles. Por ejemplo, una empresa de fabricación de automóviles, una fábrica de ordenadores, una empresa papelera o una granja de vacas.

Algunas empresas pueden asignarse a las dos categorías y otras pueden tener una orientación más de empresa de servicios que de producto, a pesar de vender los dos tipos de bienes –materiales e inmateriales al mismo tiempo–, porque hay uno que predomina sobre el otro y al revés.

Dos partes diferenciadas de los sistemas informáticos

Cada empresa tiene un sistema informático diferente. Si lo analizamos con detalle, conceptualmente y haciendo una abstracción muy generalista, podemos afirmar que en todas las organizaciones, independientemente de si están orientadas al servicio o al producto, existe una parte común y homogénea desde el punto de vista de los sistemas informáticos y otra parte muy específica, hablando de manera informática, que está muy vinculada al negocio principal de la empresa.

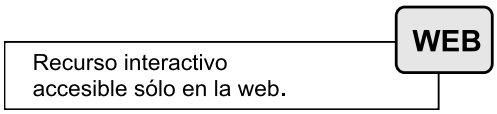
La existencia de estas dos partes informáticas, la común y homogénea y la específica, no tiene su origen en la informática o tecnología. Por el contrario, esta diferencia viene dada por el hecho de que en las empresas existen funciones o procesos comunes, que se dan en todas las empresas independientemente de su negocio –por ejemplo, los procesos relacionados con la contabilidad, la facturación, la distribución, la atención de llamadas de clientes o la relación con el proveedor. Por otro lado, la parte específica viene dada por las funciones o procesos específicos vinculados al negocio. De esta manera, los sistemas de control de proceso o de seguridad de una central nuclear son diferentes a los sistemas de control de proceso o de seguridad de un tren de alta velocidad.

Por ejemplo, un sistema informático común a la mayoría de las organizaciones, aunque sean empresas orientadas a servicios o productos, es el servidor informático con un sistema operativo orientado a red donde se almacenan los archivos (documentos, informes) que la organización produce como consecuencia de su negocio. También, como parte de un sistema informático, es común y habitual el conjunto de programas como los procesadores de texto, las hojas de cálculo o la base de datos.

Por otro lado, un ejemplo de sistemas informáticos específico muy vinculado al negocio de la organización podría ser el sistema de gestión y control de los robots de la cadena de producción de una empresa del sector del automóvil. Por el contrario, una empresa de servicios logísticos tiene un sistema informático específico orientado a su negocio, como podría ser la gestión, control y seguimiento en tiempo real de los camiones de transportes.

1.3. Ejercicio de autoevaluación

Clasificad las siguientes empresas exclusivamente como empresas de servicios o empresas de producto.



2. Caso 1. Una administración: el Ayuntamiento de VilaUOC

A continuación se presenta el caso de un ayuntamiento ficticio, pero con rasgos y características cercanos a la realidad de estas organizaciones.

2.1. Introducción funcional y organizativa

VilaUOC es una población ficticia de Cataluña con una población de unos 100.000 habitantes aproximadamente.

El Ayuntamiento de VilaUOC, que se encuentra situado en la plaza llamada Campus que está en el centro de la ciudad, se organiza y tiene las competencias que aparecen en la Ley 7/1985, de 2 de abril, reguladora de las bases del régimen local.

Organización

De manera general, esta ley establece desde el punto de vista organizativo que:

- La organización política municipal está formada por el alcalde, tenientes de alcalde, el Pleno, integrado por todos los concejales del Ayuntamiento, y la Comisión de Gobierno, formada por el alcalde y un número limitado de concejales escogidos por él.

- **El alcalde es el presidente de la corporación y algunas de sus atribuciones son las siguientes:**
  - Dirigir el Gobierno y las administraciones municipales.
  - Representar al Ayuntamiento.
  - Convocar y presidir las sesiones del pleno, de la Comisión de Gobierno y de cualquier otro órgano municipal.
  - Dirigir, inspeccionar e impulsar los servicios y obras municipales.
  - Dictar los bandos.
  - Desarrollar la gestión económica de acuerdo con el presupuesto aprobado. Ordenar pagos y rendir cuentas.
  - Ejercer la jefatura superior de todo el personal de la corporación.

- Ejercer la jefatura de la policía municipal.

- El ejercicio de las acciones judiciales y administrativas y la defensa del Ayuntamiento en las materias de su competencia.

- Adoptar personalmente, y bajo su responsabilidad, en caso de catástrofe o infortunio público o grave riesgo de los mismos, las medidas necesarias y adecuadas, y dar cuenta inmediatamente al Pleno.

- Sancionar las faltas de desobediencia a su autoridad o por infracción de las ordenanzas municipales.

- Contratar y abrir servicios dentro de unos límites prefijados por ley.

- Conceder las licencias cuando así lo dispongan las ordenanzas.

Alguna de estas responsabilidades puede delegarse a otros miembros del Gobierno municipal.

- **El Pleno, integrado por todos los concejales, lo preside el alcalde y tiene las atribuciones siguientes:**
  - El control y la fiscalización de los órganos de Gobierno.

  - La aprobación de los planes y demás instrumentos de ordenación y gestión previstos en la legislación urbanística.

  - La aprobación del Reglamento orgánico y de las ordenanzas.

  - La determinación de los recursos propios de carácter tributario, y la aprobación y modificación de los presupuestos.

  - La aprobación de las formas de gestión de los servicios y de los expedientes municipales.

  - La aceptación de las delegaciones hechas por otras administraciones.

  - La aprobación de la plantilla de personal, la relación de puestos de trabajo, las bases de las pruebas para la selección de personal y para los concursos de provisión de puestos de trabajo, la fijación de las cuantías de las retribuciones complementarias de los funcionarios y el número y regímenes del personal eventual.

  - La alteración de la calificación jurídica de los bienes de dominio público.

  - La enajenación de patrimonio.

– La votación de la moción de cesura al alcalde.

- **La Comisión de Gobierno, integrada por el alcalde y un número limitado de concejales, tiene las atribuciones siguientes:**
  – Asistencia al alcalde en sus atribuciones.

  – Las atribuciones que el alcalde u otros órganos municipales hayan delegado.

A esta organización política municipal le dan apoyo los técnicos municipales, organizados adecuadamente, como veremos más tarde.

Esta estructura política del Ayuntamiento de VilaUOC es igual al resto de los ayuntamientos del Estado español.

Diagrama organizativo municipal

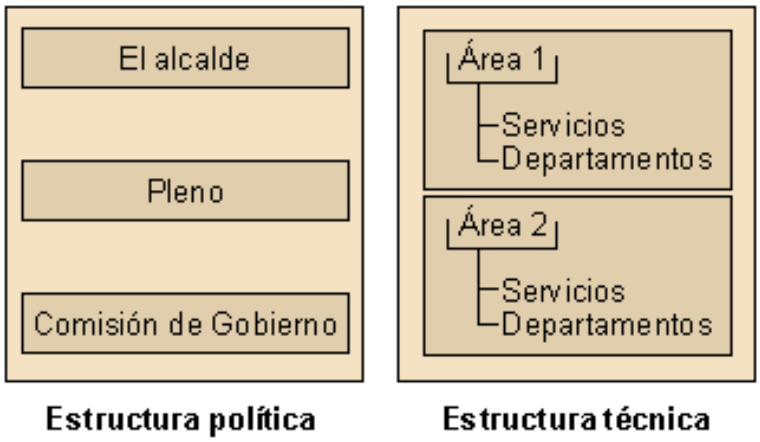


Figura 1

2.2. Funciones

Esta estructura política tiene encomendadas una serie de competencias que vienen fijadas en la Ley 7/1985, de 2 de abril, reguladora de las bases del régimen local.

De manera general, esta ley otorga las siguientes competencias a los órganos municipales:

- La seguridad en lugares públicos.

- Ordenación del tráfico de vehículos y personas en las vías urbanas.

- Protección civil, prevención y extinción de incendios.

- Ordenación, gestión, ejecución y disciplina urbanística. Promoción y gestión de viviendas, parques y jardines, pavimentación de vías públicas urbanas y conservación de los caminos y vías rurales.

- Patrimonio histórico y artístico.

- Protección del medio ambiente.

- La gestión y explotación del matadero, ferias, mercados. Defensa de los usuarios y consumidores.

- Protección de la salubridad pública.

- Participación en la gestión de la atención primaria de la salud.

- Cementerios y servicios funerarios

- Prestación de los servicios sociales y de promoción y reinserción social.

- Suministro del agua y el alumbrado público. Servicios de limpieza vial y de recogida y tratamiento de residuos, alcantarillado y tratamiento de aguas residuales.

- Protección civil, prevención y extinción de incendios e instalaciones deportivas.

- Transporte público de viajeros.

- Actividades e instalaciones culturales y deportivas. Ocupaciones del tiempo libre. Turismo.

- Participación en la programación de la enseñanza y cooperación con la Administración educativa en la creación, construcción y sostenimiento de los centros públicos docentes, participar en sus órganos de gestión y participar en la vigilancia del cumplimiento de la escolaridad obligatoria.

Para dar apoyo a sus atribuciones, el Ayuntamiento de VilaUOC dispone de una estructura técnica que está organizada por áreas. El director o máximo responsable de cada área es un político bajo el cual hay un conjunto de recursos humanos, económicos y técnicos adecuados para ejecutar las tareas encomendadas.

Concretamente, el Ayuntamiento de VilaUOC tiene ocho áreas, dirigidas por un concejal y un director de área. Cuatro de las áreas (Área de Cultura, Área de Vía Pública, Área de Promoción Estratégica y Área de Medio Ambiente) tienen una organización por servicios que, a la vez, se encuentran estructurados en departamentos, con lo cual se pueden subdividir en unidades de actuación,

según los diferentes ámbitos que abarque cada departamento. El resto de las áreas (Área de Recursos Generales, Área de Arquitectura y Urbanismo, Área de Educación y Familia y Área de Bienestar) tienen una organización por departamentos que depende de las unidades de actuación.

El esquema general de la estructura técnica es el que podemos ver al margen.

Organigrama con ocho áreas colgadas del alcalde

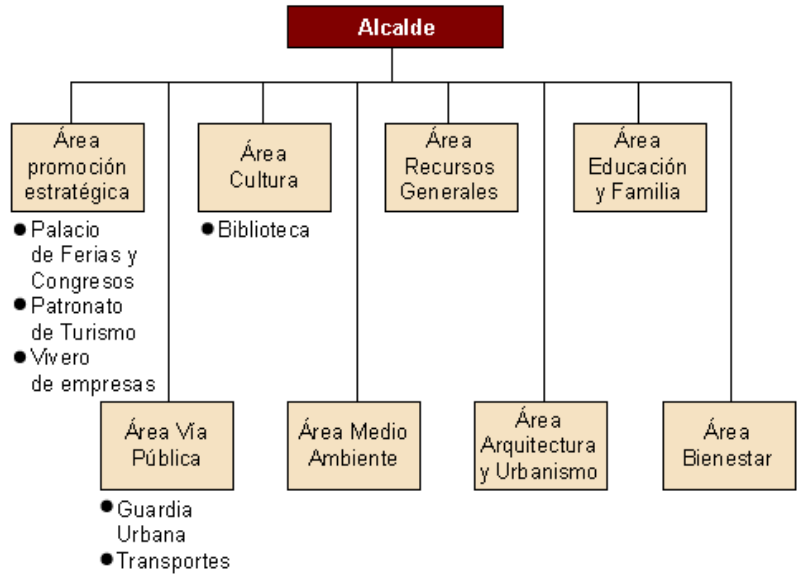


Figura 2

En los apartados siguientes se presentan con más amplitud estas áreas:

- Promoción Estratégica.
- Vía Pública.
- Cultura.

2.3. Área de Promoción Estratégica

Es el área encargada de la gestión y el desarrollo del Palacio de Ferias y Congresos de VilaUOC, del Patronato de Turismo Municipal y del vivero de empresas. Tiene la responsabilidad fundamental de detectar oportunidades económicas y sociales y desarrollarlas.

Su estructura orgánica es la siguiente: un concejal es el máximo responsable del área. La dirección ejecutiva corresponde a un director de área, que en este caso es un funcionario del Ayuntamiento.

Cada organismo –el Palacio de Ferias y Congresos, el Patronato de Turismo Municipal y el vivero de empresas– tiene un gerente que reporta directamente al director del área y que se apoya en un equipo humano especializado.

1) El Palacio de Ferias y Congresos

El **Palacio de Ferias y Congresos** se encuentra ubicado en las afueras de la ciudad (podéis ver el mapa de VilaUOC). En el mismo, se realizan los congresos y ferias programados. Los equipos informáticos del personal encargado de la administración y gestión del Palacio de Ferias, PC e impresoras, están conectados a una LAN Fast Ethernet mediante un *switch* que permite compartir información y recursos, como por ejemplo impresoras. La LAN también dispone de una pasarela de transmisión de datos sin hilos *WiFi* para dotar de conectividad en un ámbito de datos a los *stands* de determinadas ferias.

La LAN del Palacio de Ferias y Congresos dispone de una conexión con fibra óptica con el Ayuntamiento, que permite acceder a los servicios informáticos que éste proporciona desde el Área de Recursos Generales, como por ejemplo el servidor de archivo, el correo electrónico interno o correo Internet.

Red del Palacio de Ferias y Congresos

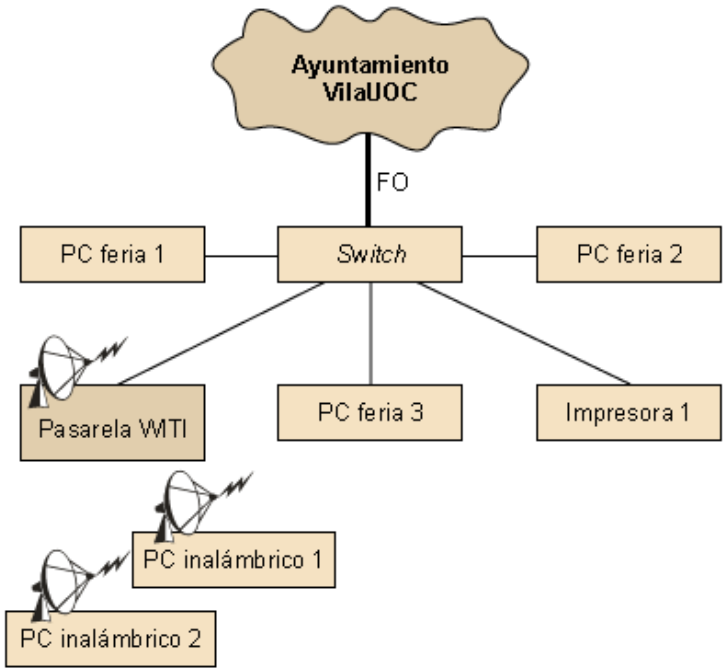


Figura 3

Sistemas y aplicaciones específicas

El Palacio de Ferias y Congresos dispone de un aplicativo que permite facilitar las gestiones relacionadas con un congreso: emisión de pases, invitaciones, control de asistencia, gestión de ponentes, elaboración de calendarios de actos y congresos, elaboración y mantenimiento de los programas de actos, gestión y control de gastos de los expositores, etc

El Palacio de Ferias y Congresos también dispone de un sistema de anulación de señales de telefonía móvil en los salones de actos y de congresos que funciona mediante un aplicativo de gestión de llamadas.

Dentro de las salas del Palacio de Ferias y Congresos no existe cobertura de telefonía móvil, con el objetivo de impedir molestias sonoras durante los actos que allí se celebran.

Cuando se entra en una sala, el propietario de un móvil tiene que introducir en un expendedor que se encuentra en la entrada de la sala su número de móvil. A continuación, se le entrega un buscapersonas con una pantalla de texto.

Automáticamente, las llamadas que recibe este móvil se desviarán a la centralita del Palacio de Ferias y Congresos, donde se atenderán. Inmediatamente después, se enviará un mensaje silencioso con un resumen de la llamada al buscapersonas que tiene el propietario del móvil para que esté informado.

Por otro lado, el Palacio de Ferias y Congresos dispone de una aplicativo de gestión de interiores que permite configurar la disposición más óptima de los espacios de los *stands* de una feria determinada.

Hay algunos servicios informáticos que son proporcionados directamente desde el Ayuntamiento. Concretamente, el Servicio de Informática del Ayuntamiento de VilaUOC proporciona al Palacio de Ferias y Congresos de VilaUOC el acceso a Internet, sistema de compartición de archivos, correo interno y externo (Internet), gestión de contabilidad, sistema antivírico, sistema de copias de seguridad, etc.

2) El Patronato de Turismo Municipal de VilaUOC

El Patronato de Turismo Municipal de VilaUOC tiene dos ubicaciones. Una, la sede administrativa y de gestión situada en un edificio emblemático de la ciudad, cerca del centro; y otra, una oficina de atención al público sobre información turística, situada en el centro de la ciudad y en un módulo prefabricado, debido a su provisionalidad mientras se decide su ubicación definitiva (podéis ver el mapa de VilaUOC).

La sede central del Patronato Municipal de Turismo está conectada al Ayuntamiento, mientras que la oficina provisional de atención al público sólo está conectada a Internet mediante ADSL.

Existe una LAN Ethernet en cada ubicación, con un total de cinco PC en la sede central y dos PC en la oficina de atención al público, más algunas impresoras de tecnología de tinta en color y láser monocromo. La LAN de la sede central utiliza un *switch*, mientras que la oficina de atención al público tiene un HUB con ocho puertos.

Red del Patronato Municipal de Turismo

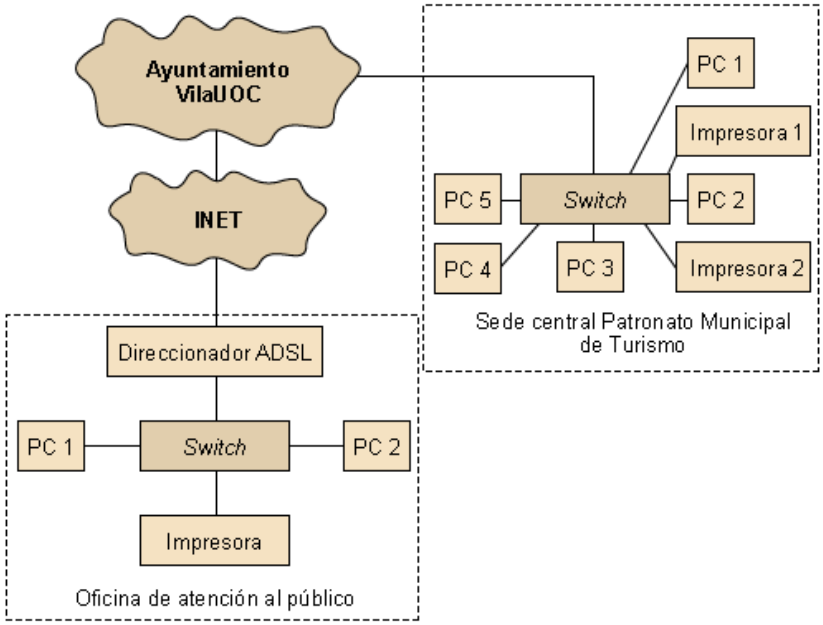


Figura 4

Las dos oficinas se intercambian información mediante el correo electrónico, enviando los archivos o documentos anexados a un mensaje.

Sistemas y aplicaciones específicas

El Patronato Municipal de Turismo tiene toda la información turística del municipio publicada en Internet. De esta manera, cuando la oficina de atención al ciudadano tiene que dar una información presencialmente en su oficina, accede a la Web y hace la busca adecuada utilizando la misma fuente de información y las mismas herramientas de las que pueden disponer los internautas.

El Patronato Municipal de Turismo tiene un aplicativo con el que gestiona las consultas realizadas mediante diferentes canales de entrada y sus respuestas.

Hay que destacar que la oficina de atención presta sus servicios de información mediante varios canales: teléfono, el correo electrónico y el tablero.

Este aplicativo permite posteriormente extraer datos e indicadores, como el numero de consultas por comunidad o país realizadas, los ámbitos turísticos de mayor interés, etc., para posteriormente tomar decisiones.

En este aplicativo, cualquier consulta está asociada a unos datos personales de la persona que la realiza. También se almacena el texto completo de la consulta y el texto completo de la respuesta, y se categoriza la consulta.

Incluso si se ha facilitado el correo electrónico a la hora de hacer la consulta, el patronato municipal de turismo genera y envía un mensaje a través de Internet pasados siete días de la consulta, con un texto prefijado en el que se dan las gracias por la visita a la ciudad de VilaUOC y se desea que la vuelta a casa haya sido excelente.

El Patronato Municipal de Turismo también recibe servicios informáticos del Área de Recursos Generales del Ayuntamiento.

Concretamente, el Servicio de Informática del Ayuntamiento de VilaUOC proporciona a la sede central del Patronato Municipal de VilaUOC el acceso a Internet, sistema de compartición de archivos, correo interno y externo (Internet), gestión de contabilidad, sistema antivírico, sistema de copias de seguridad, etc

El Servicio de Informática del Ayuntamiento de VilaUOC proporciona a la sede provisional de atención al público del Patronato Municipal de Turismo únicamente servicios de apoyo y mantenimiento de la infraestructura informática.

3) El vivero de empresas

El **vivero de empresas** se encuentra en las afueras de VilaUOC, dentro de un polígono industrial (podéis ver el mapa de VilaUOC). Consta de una nave en la cual se ha compartimentado el espacio para dar cabida a empresas de nueva creación y para facilitar su crecimiento inicial. Concretamente, el vivero presta y facilita una serie de servicios de apoyo empresarial orientados a mejorar las condiciones de desarrollo de las empresas durante sus primeras etapas operativas, con el objetivo de incrementar sus posibilidades de supervivencia.

Algunos de estos servicios y apoyo son, por ejemplo, la recepción y atención de llamadas de los clientes y proveedores de las empresas que allí se encuentran, la gestión y liquidación de impuestos, etc. De momento, no tienen conexión con el sistema informático del Ayuntamiento y únicamente disponen de una red local "switcheada" con una VPN (*Virtual Private Network*) para cada empresa y una VPN común, a la que se conectan los sistemas informáticos comunes. La LAN del vivero tiene conectado un ADSL de 512 Mbps que permite el acceso a Internet de las empresas del vivero.

Red vivero de empresas

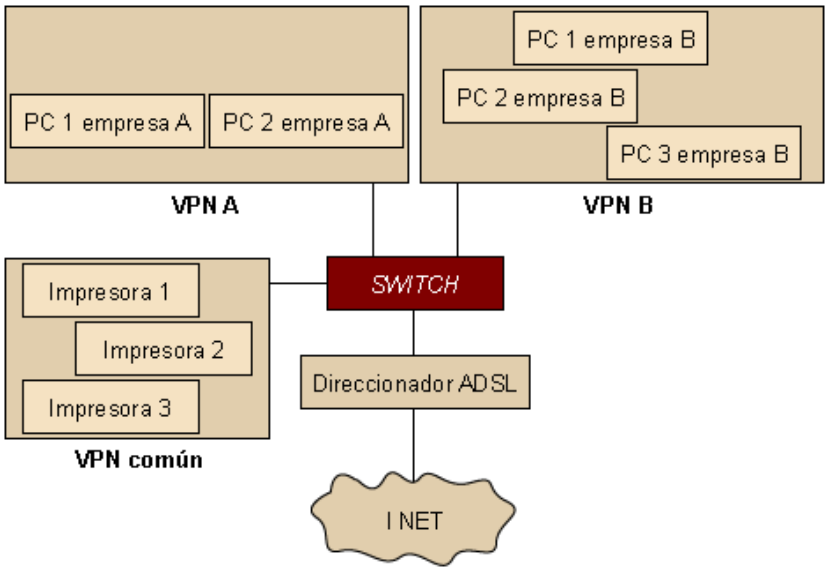


Figura 5

Todos los sistemas y aplicaciones del tipo común que necesitan las empresas los provee el propio vivero, y son independientes de los sistemas informáticos del Ayuntamiento de VilaUOC.

Las aplicaciones muy específicas son adquiridas, explotadas y mantenidas por la propia empresa.

El vivero tiene contratado el apoyo informático básico –para sí mismo y para las empresas a las que da servicios– a una empresa de servicios informáticos de VilaUOC.

2.4. Área de Vía Pública

Es el área encargada de la seguridad en lugares públicos, de la ordenación del tráfico de vehículos y personas en las vías urbanas, de la protección civil, prevención y extinción de incendios y del transporte público, entre otras funciones.

Su estructura orgánica es la siguiente: un concejal es el máximo responsable del área. La dirección ejecutiva corresponde a un director de área, que en este caso se trata de un cargo de confianza de libre designación por parte del alcalde.

Los dos servicios mayores dentro de esta área son la Guardia Urbana y el Servicio de Transporte Municipal. Estos dos servicios están dirigidos por un inspector jefe, en el caso de la Guardia Urbana, y por un gerente en el caso del Servicio de Transporte Municipal.

1) Guardia Urbana

La Guardia Urbana se encuentra en un edificio independiente del Ayuntamiento, en el primer cinturón (rondas) de VilaUOC (podéis ver el mapa de VilaUOC). El parque móvil de vehículos, así como las oficinas de gestión y administración, se encuentran juntas en el mismo lugar.

Dispone de una LAN en Gigabit Ethernet, con todos los equipos informáticos conectados, incluso los teléfonos. Esta LAN está conectada con el Ayuntamiento mediante un enlace de fibra óptica a Gigabit Ethernet.

Red Guardia Urbana

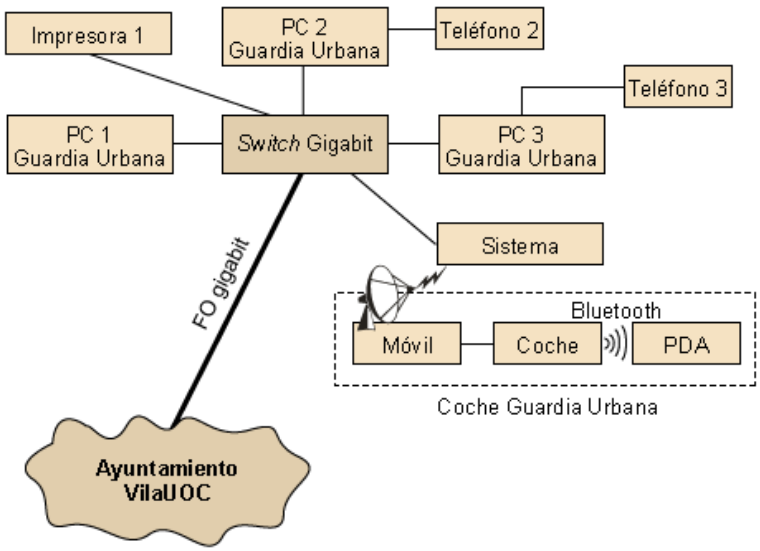


Figura 6

Sistemas y aplicaciones específicas

Es el único centro, junto con el Ayuntamiento, que utiliza la red de datos para transmitir voz mediante un sistema de telefonía denominado *VoIP* (voz sobre IP).

Todas las llamadas que se realizan y se reciben en la Guardia Urbana utilizan, por un lado, la centralita de telefonía del Ayuntamiento y, por otro lado, la red de datos que interconecta el Ayuntamiento y la Guardia Urbana.

Desde todos los ordenadores de la Guardia Urbana, y mediante unos auriculares y un micrófono, se pueden realizar o recibir llamadas telefónicas. Éste es un sistema experimental que se está probando para evaluar posteriormente su implantación en todos los centros del Ayuntamiento.

Además, la Guardia Urbana dispone de un sistema de localización y seguimiento de vehículos policiales en tiempo real, lo que permite asignar servicios policiales óptimamente. El sistema está formado por dos partes: el sistema instalado en el vehículo policial y el sistema central instalado en la sede de la Guardia Urbana.

El sistema de localización y seguimiento que se encuentra en el vehículo está formado por un receptor GPS, un terminal de telefonía móvil y un reducido equipo informático que tiene una conexión sin hilos de corta cobertura mediante tecnología Bluetooth.

El sistema central consta de varios terminales de telefonía móvil conectados a un equipo informático. Este sistema central es asimétrico, es decir, recibe más información de los vehículos que la que envía.

Cada cinco segundos, gracias al sistema GPS, el sistema ubicado en el vehículo envía su posición (latitud y altitud) a la central mediante una comunicación permanente (*always on*) de telefonía móvil GPRS.

El sistema informático de la central de la Guardia Urbana recoge los datos. Éstos son procesados para a continuación mostrarlos en un mapa grande del centro de mando de la Guardia Urbana, en el que se ubica cada coche en su correspondiente posición geográfica.

Multas sin hilos

Los agentes de la Guardia Urbana disponen en el vehículo policial de un PDA (*Personal Digital Assistant*) con un aplicativo que permite introducir y gestionar parcialmente las infracciones del Código de Circulación o de otras normas de la vía pública.

Una vez los agentes han introducido los datos de la infracción, ésta es enviada del PDA al pequeño equipo informático del vehículo mediante una conexión *bluetooth*, que hemos visto antes. Desde el equipo informático, se enviará a la central de la Guardia Urbana mediante una transmisión móvil con GPRS.

Finalmente, en la central se procesarán los datos y se enviarán al infractor por correo ordinario.

Identificación de personas

Los PDA también sirven como apoyo a las tareas de identificación de persona.

Cuando la Guardia Urbana identifica a una persona y quiere saber si existe una orden de busca y captura, introduce el DNI en el PDA y recibe al cabo de unos segundos la información de la central. Esta comunicación se establece de la misma manera que la de la gestión de multas.

De la misma manera, también puede introducir las matrículas de coches para saber si se encuentra robado, denunciado, abandonado o le falta el pago de algún impuesto o tasa.

2) Empresa de transportes

La empresa municipal de transportes gestiona el servicio de autobuses urbanos e interurbanos, este último mediante una concesión administrativa por parte de la Generalitat.

Dispone de un único edificio en las afueras de la ciudad, donde se encuentran las oficinas administrativas y de gestión y el parque móvil y taller de los autobuses (podéis ver el mapa de VilaUOC).

Ha conseguido hace muy poco la certificación de calidad ISO 9002-2001, por la cual uno de los compromisos adquiridos más relevantes es la puntualidad en el servicio de autobuses.

Dispone de una red de datos metropolitana peculiar que fue diseñada e implementada hace cinco años. En cada parada de autobuses de la ciudad se dispone de un pequeño equipo informático, un simple PC, conectado con una línea de teléfonos (RTC) mediante un módem a 56 kbps.

Los sistemas informáticos de las paradas de autobuses envían y reciben datos de los sistemas informáticos centrales de la empresa de transporte llamando a un teléfono de tarificación local.

El sistema central está formado por un servidor con un gestor de base de datos relacional conectado a una red local LAN Ethernet. Esta red local está conectada, mediante un sistema LMDS (*Local Multipoint Distribution System*), con el Ayuntamiento de VilaUOC.

Red empresa de transportes

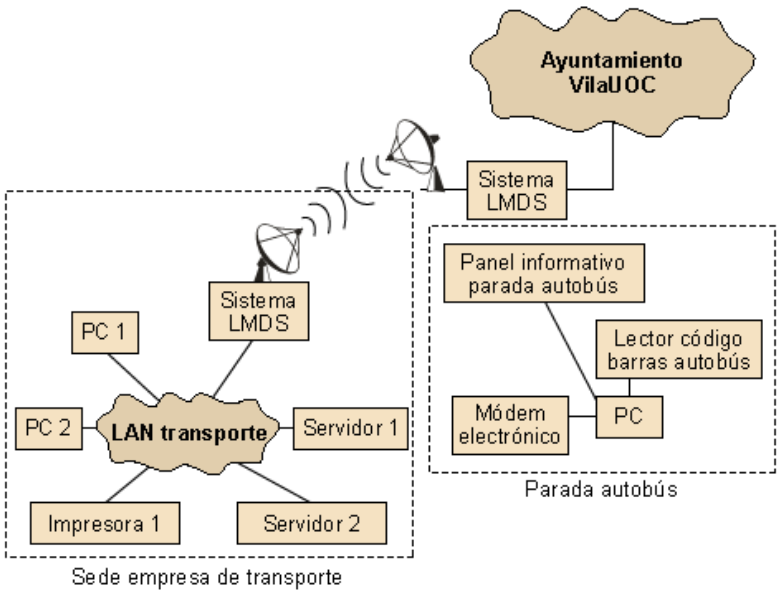


Figura 7

Sistemas y aplicaciones específicas

Uno de los sistemas de información más antiguos, pero a la vez más innovadores, de la empresa de transportes es la gestión del tiempo y movilidad de los autobuses.

Cada autobús está dotado de un código de barras pequeño en su lateral derecho. Cada vez que entra en una parada se lee con un lector láser el código de barras que lleva el autobús, y el sistema informático que se encuentra en la parada envía a la central el código leído mediante una conexión RTC. La información es procesada y enviada de nuevo a los paneles informativos de las paradas, y se indica una previsión del tiempo de llegada del siguiente autobús.

2.5. Área de Cultura

El Área de Cultura es la encargada de la promoción y defensa del patrimonio cultural. El Área de Cultura debe crear y mantener relaciones de colaboración con las entidades culturales del municipio, y tiene que divulgar y promover la cultura en todos los ámbitos, principalmente en todo aquello que haga referencia al ámbito local.

El Área de Cultura del Ayuntamiento de VilaUOC dispone de una biblioteca pública cerca del parque Ferrater. La biblioteca tiene como prioridad permitir la consulta y préstamo de material como libros, vídeos, CD y DVD. Sin embargo, también ofrece espacios para estudiar, reunirse y consultar Internet gratuitamente. Por este motivo, dispone de espacios diferenciados como la biblioteca, la estudioteca, la Interneteca y la encuentroteca.

La biblioteca dispone de una red local segmentada por un *switch* en cinco zonas diferenciadas: la red administrativa a la que están conectados los PC de los trabajadores de la biblioteca, y las redes de la biblioteca, estudioteca, Interneteca y encuentroteca.

Esquema conceptual de biblioteca

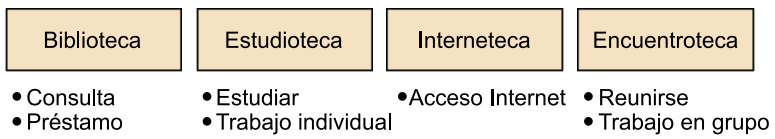


Figura 8

Tienen un servidor de archivos donde, además, residen los aplicativos específicos de la biblioteca que veremos más adelante.

La biblioteca tiene dos conexiones hacia el exterior: una conexión en un ámbito de datos con el Ayuntamiento y un enlace a Internet mediante un ADSL de 2 Mbps.

La conexión con el Ayuntamiento se utiliza para el intercambio de información administrativa y para acceder a los servicios que el Servicio de Informática del Área de Recursos Generales provee, como por ejemplo Internet, correo electrónico interno y externo (Internet), etc.

La conexión a Internet mediante un ADSL es utilizada, por razones de seguridad, exclusivamente por los usuarios de la Interneteca. De esta manera, el tráfico administrativo de datos de los trabajadores viaja por una línea y el tráfico en Internet de los usuarios de la Interneteca, por otra.

Red Biblioteca

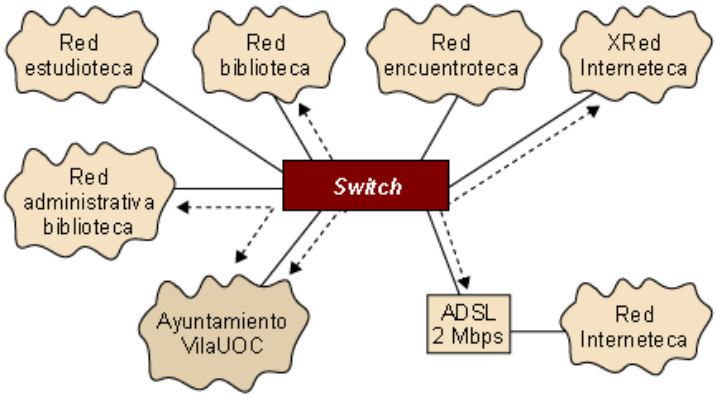


Figura 9

Sistemas y aplicaciones específicos

Para acceder a la Interneteca y a la encuentroteca, se recomienda hacer previamente una reserva. Si no se ha reservado el día y hora, se podrán utilizar los servicios siempre que haya libre algún recurso y mientras no sea utilizado por alguien que lo haya reservado.

La reserva se puede realizar presencialmente en la biblioteca, utilizando un terminal específico para esta función, o no presencialmente, mediante Internet.

La estudioteca es un espacio de libre acceso y utilización sin, de momento, ninguna norma asociada.

La biblioteca permite la consulta de su material disponible, pero también posibilita el préstamo de la mayoría del material en formatos papel, CD, DVD, etc.

Funcionalmente, la consulta y préstamo tienen el mismo carácter, las dos acciones son préstamos y sólo se diferencian en dos cosas: el tiempo que pasa entre que se toma un material y se devuelve, y el espacio por donde se mueve el objeto prestado. Una consulta tiene un tiempo de préstamo muy corto, como mucho todo el día mientras permanece abierta la biblioteca. El objeto de consulta nunca sale de la biblioteca.

Por el contrario, un préstamo se realiza por distintos días y el objeto prestado sale de la biblioteca.

La consulta y el préstamo se encuentran informatizados a modo de autoservicio mediante un sistema informático. Cuando un usuario quiere un material para consultar dentro de la biblioteca o para llevárselo a casa mediante un préstamo, hay un equipo que permite desactivar la seguridad activa de este elemento consultable o prestable, después de introducir nuestra tarjeta personal con banda magnética y de leer el código de barras del objeto que desea llevarse o consultar.

De esta manera, la consulta y el préstamo se realizan únicamente con la intervención humana del usuario de la biblioteca y, además, conseguimos tener un registro informatizado de todo el material que se consulta y a la vez de todo el material que se presta.

2.6. Anexo mapa de VilaUOC

Mapa de la ciudad de VilaUOC

- **El Palacio de Ferias y Congresos** se encuentra ubicado en las afueras de la ciudad.

- **El vivero de empresas** se encuentra en las afueras de VilaUOC, dentro de un polígono industrial.

- **El Patronato de Turismo Municipal de VilaUOC** tiene dos ubicaciones. Una, la sede administrativa y de gestión situada en un edificio emblemático de la ciudad, cerca del centro; y otra, una oficina de atención al público sobre información turística, situada en el centro de la ciudad y en un módulo prefabricado, debido a su provisionalidad mientras se decide su ubicación definitiva.

- La **Guardia Urbana** se encuentra en un edificio independiente del Ayuntamiento, ubicado en el primer cinturón (rondas) de VilaUOC.

- La **empresa de transporte municipal** dispone de un único edificio a las afueras de la ciudad, donde se encuentran las oficinas administrativas y de gestión y el parque móvil y taller de los autobuses.

- VilaUOC dispone de una **biblioteca pública** cerca del parque Ferrater.

- El **Área de Educación y Familia** se encuentra ubicada en la sede central del Ayuntamiento.

- El **Ayuntamiento** de VilaUOC, que se encuentra situado en la plaza denominada Campus en el centro de la ciudad.

3. Caso 2. Una empresa de fabricación de producto. MecanoUOC

A continuación se presenta el caso de una empresa ficticia, pero con rasgos y características cercanos a la realidad de estas organizaciones.

3.1. Introducción funcional y organizativa

MecanoUOC es una empresa que se dedica a fabricar y distribuir ordenadores PC.

Concretamente, MecanoUOC adquiere las piezas que forman un ordenador, normalmente en mercados extranjeros. Las ensambla en la fábrica que tiene en VilaUOC, y posteriormente distribuye y vende sus PC por medio de las tiendas franquiciadas en un ámbito nacional o mediante un portal de comercio electrónico que tiene en Internet.

Actualmente, el mercado de MecanoUOC es el Estado español, pero no se descarta una internacionalización si los crecimientos en facturación y beneficio se mantienen como hasta ahora.

MecanoUOC está situada en el polígono industrial de VilaUOC, y toda la organización se encuentra centralizada en una misma localización física que está compuesta por un edificio, un almacén y una nave donde se encuentra únicamente la cadena de producción de PC.

MecanoUOC tiene actualmente 1.200 trabajadores, que están adscritos a uno de los cinco departamentos que constituyen la empresa:

- El Departamento de Investigación, Desarrollo e Innovación.
- El Departamento de Marketing.
- El Departamento de Producción.
- El Departamento Comercial, Logístico y de Expansión.
- El Departamento de Administración, Económico y Financiero.

Los cinco departamentos están dirigidos por una unidad denominada Gerencia, departamento que está integrado por un equipo de alta dirección que dirige a toda la organización.

Esquema organigrama MecanoUOC

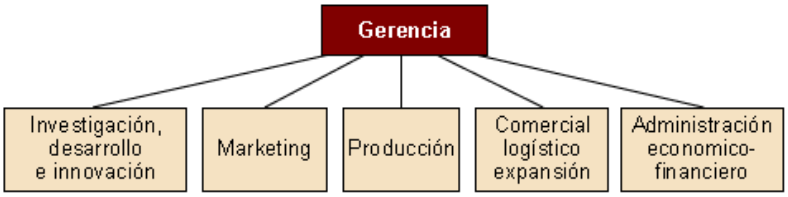


Figura 10

Aunque existe un organigrama clásico jerárquico como el que representa el esquema superior, el modelo de gestión de MecanoUOC está basado en una gestión, dirección y organización funcional por unidades de negocio, de tal manera que una unidad de negocio puede utilizar recursos de dos o más departamentos.

Para llevar a cabo su misión, MecanoUOC tiene algunas funciones consideradas estratégicas, algunas de las cuales las veremos desde un punto de vista funcional y de sistemas informáticos. En concreto, trataremos las funciones siguientes:

- Recursos humanos.
- Logística y distribución de producto.
- Facturación y contabilidad.
- Gestión de franquicias.
- Marketing.
- Informática.

3.2. Recursos Humanos

La gestión de los recursos humanos en MecanoUOC es estratégica y al mismo tiempo peculiar por varias razones.

MecanoUOC se encuentra situada en una región donde la mano de obra no es abundante. Además, tiene una plantilla de una media de edad muy joven y, por otro lado, el negocio de la informática cambia muy rápidamente y provoca que los conocimientos del personal se tengan que actualizar constantemente. Por último, MecanoUOC tiene un índice muy alto de bajas voluntarias en los puestos de trabajo que no requieren de personal cualificado (mozos de almacén, operarios, etc).

El Departamento de Recursos Humanos tiene que velar no sólo por captar y contratar a los mejores trabajadores, sino también por mantenerlos, formarlos y conseguir una implicación total del trabajador hacia la empresa.

Por estas razones, todos los currículos que recibe MecanoUOC son almacenados en un sistema informático y toda la información relativa a un proceso de selección también es conservada por si existen necesidades futuras y no planificadas de contratación de personal.

Sistemas y aplicaciones específicas

MecanoUOC dispone de una aplicativo de gestión de personal para llevar a cabo todas las tareas propias de un Departamento de Recursos Humanos. Este aplicativo tiene como apoyo una base de datos relacional.

Dentro del aplicativo, constan los datos personales y profesionales de los trabajadores, así como un historial en continua actualización.

En este historial constan las retribuciones adjudicadas mensualmente, así como la formación general y específica cursada, la evaluación anual realizada por el jefe, los subordinados –si se trata de un mando– y otros compañeros en línea horizontal jerárquica (sistema de evaluación de personal denominado *evaluación 360º*). Dentro del historial, también podemos encontrar información relativa a las ayudas y beneficios que MecanoUOC tiene y da a sus trabajadores, como las ayudas para los hijos de los trabajadores menores de dieciocho años, las ayudas relativas a la salud como monturas de gafas y cristales focales u otros.

Los trabajadores tienen acceso a toda la información almacenada dentro del aplicativo de gestión de personal mediante una interfaz denominada *Portal del empleado*.

Este portal es un sistema en el que, mediante una validación personal, aparece en un navegador información relacionada con el trabajador y MecanoUOC.

El portal saca la información de la base de datos relacional que utiliza el aplicativo de gestión de personal, y la representa en un navegador con una ordenación orientada a informar al trabajador de los últimos datos introducidos o calculados, como la nómina.

Mediante el Portal del empleado, y como ejemplo, se puede acceder a los detalles de la información de las últimas nóminas cobradas, consultar el estado de las ayudas solicitadas, imprimir un certificado de la formación realizada o consultar el resultado alcanzado en la evaluación personal anual de 360º.

El sistema de gestión de personal no es un aplicativo aislado dentro de la informática corporativa de MecanoUOC. Este aplicativo se encuentra conectado a distintos sistemas informáticos corporativos, como por ejemplo la contabilidad de MecanoUOC, el sistema de acceso telemático a Hacienda o el sistema de acceso telemático de la Tesorería de la Seguridad Social.

3.3. Logística y distribución del producto

La distribución es parte de la logística operacional en MecanoUOC.

La distribución es la parte que se encarga de hacer llegar los paquetes al destino en la fecha y hora comprometidas.

MecanoUOC apuesta por no tener un *stock* de ordenadores, sino por trabajar bajo demanda con un tiempo de respuesta muy corto. De este modo, intenta acortar al máximo posible el tiempo de producción, distribución y entrega del producto una vez el cliente lo ha pedido.

La idea de MecanoUOC es vender exactamente el PC que quiere el cliente, con las características y elementos que éste decida a la hora de comprarlo.

MecanoUOC se diferencia en este punto de la competencia que tiene exclusivamente un número limitado de modelo. Cuando se compra un ordenador mediante las franquicias de MecanoUOC o por medio de Internet, el cliente decide qué caja quiere, qué placa madre, qué disco duro, qué memoria, qué DVD, qué módem, qué tarjeta de sonido, qué tarjeta de red, cuántos puertos USB, qué monitor, etc.

Una vez realizada la elección por parte del cliente de todos los elementos que tendrá un ordenador, el pedido se recepciona a los sistemas informáticos de la central de MecanoUOC y a partir de este momento empieza el montaje del ordenador pedido.

Una vez montado el ordenador, MecanoUOC aplica un test de calidad al PC denominado *Boom Test*, que consiste en dejarlo funcionando y realizando cálculos intensivos durante unas horas, dentro de una sala donde se produce una variación forzada de la temperatura de ambiente con el objetivo de analizar las reacciones de las diferentes partes del PC.

Si el PC supera el test, automáticamente es embalado e introducido en el camión adecuado según el destino final.

Diagrama funcional pedido ordenador

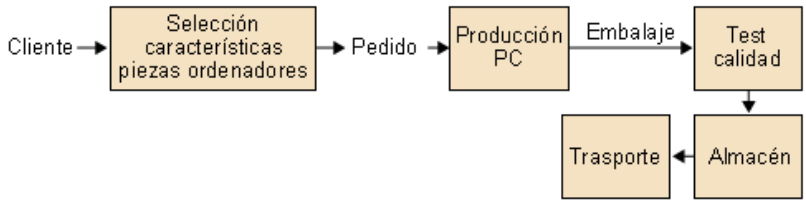


Figura 11

Sistemas y aplicaciones específicas

El sistema informático que da apoyo al proceso de producción y distribución de PC es complejo.

Los pedidos son recibidos por el mismo sistema informático, tanto si son introducidos mediante Internet o mediante una tienda franquiciada.

Una vez recibido el pedido, el sistema informático actúa sobre los mecanismos de la cadena de producción y abre la puerta del depósito que contiene cada elemento solicitado en el pedido (memorias, disco, caja torre PC, DVD, CD, etc.).

Todos los elementos caen encima de una cinta transportadora y son dirigidos hasta un puesto de trabajo atendido por una persona, la cual hará el ensamblaje manualmente.

Todos los elementos que constituyen un ordenador, como por ejemplo las memorias, los discos duros, los puertos USB, etc., se encuentran identificados mediante la utilización de un código de barras.

Una vez ensambladas las piezas y montado el PC, el operario asigna un nuevo código de barras al ordenador que actuará de identificador, y que quedará registrado en el sistema informático. Entonces, se establecerá una relación entre este identificador del ordenador y todos los identificadores de las piezas que lo forman.

A continuación, y mediante un toro automatizado, se dirigirá el PC montado hasta la sala del *Boom Test*, donde unos brazos robotizados pondrán una serie de sondas que estarán conectadas a un sistema de control y medición.

Después de que haya pasado el tiempo establecido en los procedimientos de calidad, y si los sensores de control y medición no han detectado ninguna anomalía, este sistema comunicará al sistema informático de gestión de pedidos que el PC se encuentra en condiciones de ser entregado.

A continuación, el sistema informático programará el toro automatizado para transportar el PC de la sala del Boom Test al sistema de embalaje. Una vez embalado, se pondrá el PC en el almacén junto con los otros PC que deben ser cargados en el mismo camión.

Las comunicaciones entre el sistema informático central y los toros automatizados se realizan con un sistema sin hilos. MecanoUOC tiene distribuidas por la nave industrial donde se encuentra la cadena de producción y en el almacén una serie de antenas que permiten transmitir datos sin hilos entre el sistema informático central y los toros automatizados.

El sistema informático de gestión de pedidos calcula desde el primer momento en el que se hace el pedido en qué camión será transportado el PC para llegar al cliente. El sistema informático también calcula la hora de salida de cada camión, la ruta que hay que seguir, el tiempo y las paradas que se deben realizar tanto para descargar material como para que descanse el conductor o conductores, si el trayecto es muy largo.

La carga del camión la hará un toro automatizado programado por el sistema informático.

El sistema informático también avisa con antelación al conductor o conductores del camión de la hora de salida prevista.

MecanoUOC sigue casi en tiempo real sus transportes.

Cada camión está equipado con un receptor GPS, un terminal de telefonía móvil y un pequeño sistema informático. Cada cinco minutos se realiza una lectura de la latitud y longitud mediante el receptor GPS, y estos datos son enviados mediante un mensaje SMS a la sede central.

De manera automática, todos los pedidos que se encuentran en aquel camión quedan posicionados geográficamente en el sistema informático.

En todo momento el cliente puede consultar el estado de su pedido mediante Internet, introduciendo el código del pedido y una clave secreta que se ha entregado en el momento de introducir el pedido en el sistema informático. El cliente también puede consultar en todo momento el estado de producción y distribución de su PC. Igualmente, es posible consultar el lugar geográfico donde se encuentra el PC mientras el camión de transporte lo lleva al destino.

Pedido

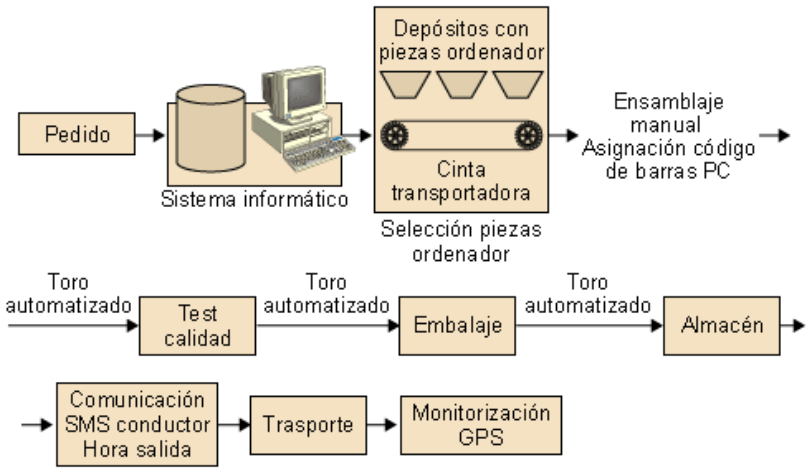


Figura 12

3.4. Facturación y contabilidad

Una vez un producto es embalado en MecanoUOC después de realizar la gestión del pedido, como hemos visto en el apartado "Logística y distribución del producto", y antes de introducirlo en el camión de transporte para entregarlo al cliente, se inserta dentro del sistema informático contable el correspondiente apunte contable, se genera una factura y se imprime en papel.

Para generar este documento, es necesario extraer los datos del sistema de pedidos en el que aparece el cliente, el producto o productos solicitados y el precio asignado.

La factura generada tiene un vencimiento asignado por defecto de sesenta días, pero hay determinados clientes, algunas franquicias y algunos grandes clientes cuyo periodo de vencimiento asignado es de noventa días.

El sistema de contabilidad se encuentra conectado al sistema de consultas bancarias de los bancos mediante los que opera MecanoUOC.

Cuando se realiza el pago de una factura mediante una transferencia bancaria a la cuenta de MecanoUOC, el sistema de contabilidad cambia el estado de la factura de "Pendiente de cobro" a "Cobrado".

Si se supera el periodo de vencimiento de una factura sin que se haya cobrado, el sistema de contabilidad imprime una carta tipo aviso que será enviada al cliente recordando que si no se produce el pago de la factura pendiente en un plazo de diez días, se generará una nueva factura con un recargo en concepto de intereses por no haber abonado la primera.

Este proceso se repite hasta que se abona el saldo pendiente.

El sistema de contabilidad suministra información al programa ATD (sistema de Ayuda a la Toma de Decisiones, que veremos más tarde), como por ejemplo el total facturado a una franquicia durante un periodo de tiempo determinado, las deudas pendientes por franquiciado, el cumplimiento de los plazos de pago y otras informaciones necesarias para conocer y evaluar las relaciones entre MecanoUOC y sus franquiciados.

3.5. Gestión de franquicias

MecanoUOC utiliza las franquicias como mecanismo de presencia y expansión territorial, y como medio de relación comercial con el cliente final.

Por esta razón, tiene todo un equipo humano dedicado a las gestiones relacionadas básicamente con la captación y análisis de nuevos franquiciados y mantener a los franquiciados actuales.

El Departamento de Franquicias requiere acceso a fuentes de información externa con el objetivo de analizar la viabilidad de conceder una nueva franquicia a una persona o a una sociedad.

Al mismo tiempo, requiere información externa con el fin de analizar la viabilidad de implantación de una franquicia en un determinado territorio (PIB de la zona, número de habitantes, renta per cápita, competencia directa e indirecta, etc).

Esta información es recogida e introducida en un sistema informático que realiza cálculos y que determinará, mediante un sistema experto, la viabilidad y garantías de conceder una nueva franquicia a una persona o sociedad, y la viabilidad de ubicar el negocio en el lugar propuesto por el interesado.

Por otro lado, el Departamento de Franquicias necesita y conoce en todo momento el estado de la relación entre MecanoUOC y el franquiciado para actuar proactivamente en la resolución de problemas o incidencias de cualquier tipo: cobros, facturación, plazos de entregas, formación, etc.

Sistemas y aplicaciones específicas

El Departamento de Franquicias tiene un sistema informático de nueva implantación que actúa como un *data warehouse*, donde se recoge de manera selectiva toda la información necesaria vinculada con las relaciones entre los franquiciados y MecanoUOC.

De esta manera, el sistema permite saber en todo momento y de manera numérica, con una escala de 0 a 10, cuál es la valoración que tiene MecanoUOC de un franquiciado y cuál es la valoración que tiene un franquiciado de MecanoUOC.

El sistema está formado por una base de datos relacional que se conecta, por medio de un estándar XML de intercambio de datos, con los otros sistemas corporativos, como por ejemplo:

- Contabilidad y facturación, para saber el estado de pago de los pedidos enviados al franquiciado y crecimientos y decrecimientos de la facturación.

- Gestión de pedidos, para saber el tipo de productos que pide el franquiciado y el cumplimiento del tiempo de entrega por parte de MecanoUOC.

Esquema conceptual evaluación relación

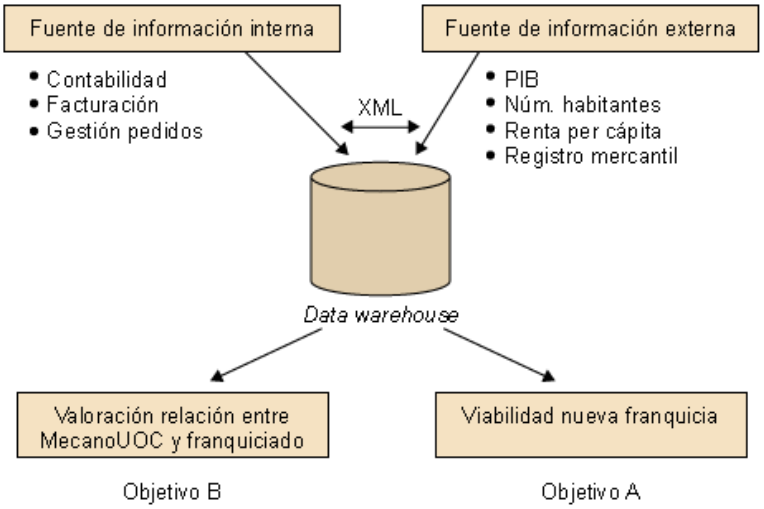


Figura 13

3.6. Marketing

Una de las misiones del Departamento de Marketing de MecanoUOC consiste en controlar a la competencia y a sus proveedores.

El Departamento de Marketing quiere saber en tiempo real qué tiene la competencia (PC, características, precios, etc.) y qué novedades saca al mercado.

Por otro lado, el Departamento de Marketing necesita estar informado en tiempo real de información generada por sus proveedores. Por ejemplo, MecanoUOC tiene interés en conocer las características y precios de las piezas que le suministran los proveedores y que sirven para montar un ordenador, y también le interesa conocer las novedades que los proveedores sacan al mercado.

Toda esta información es gestionada por un sistema ATD (Ayuda a la Toma de Decisiones).

Con esta información, el Departamento de Marketing puede realizar la adquisición de las piezas que necesita que tengan menor precio, puede copiar a la competencia imitando el último modelo que saque al mercado, puede innovar incorporando nuevas piezas a los ordenadores que salen al mercado, etc

Sistemas y aplicaciones específicas

El ATD tiene tres bloques funcionales: un robot, una base de datos documental y un módulo de toma de decisiones.

El robot es un programa informático que accede a determinadas páginas web, captura información pública de las mismas y la almacena en la base de datos documental.

El robot contiene un conjunto de URL (*Uniform Resource Locator*) o lo que es lo mismo, direcciones de web; por ejemplo, la página web en la que aparecen las novedades de la competencia.

El robot accede a las páginas web cada cinco minutos, las veinticuatro horas, y compara la información que hay en la base documental con la que se encuentra en Internet. Si la página almacenada y la que se encuentra en Internet son diferentes, entonces hay un cambio de contenido y el robot descarga la nueva página y la almacena en la base de datos documental.

La base de datos documental es el depósito de información del robot y, a la vez, es la fuente de información del módulo de toma de decisiones.

Cada vez que se produce un cambio en alguna de las URL monitorizadas en el sistema, el robot envía un mensaje al módulo de toma de decisiones y éste recalcula la decisión óptima con la nueva información; es decir, valida si las decisiones tomadas anteriormente con las antiguas condiciones se mantienen, o si da nuevas instrucciones y decisiones que hay que tomar siguiendo una serie de reglas establecidas y un aprendizaje continuo.

Por ejemplo, un cambio de precio al alza de una pieza adquirida por MecanoUOC para montar sus PC puede dar como resultado, dentro del ATD, recomendar la adquisición de una pieza de otro proveedor de similares características pero con un precio de adquisición más bajo. El ATD también puede recomendar introducir una nueva pieza en el montaje de los PC si mejora los resultados finales y el coste final de producción no se ve incrementado en un límite preestablecido.

En lo que respecta a la competencia, el sistema ATD puede alertar de la existencia de un producto nuevo en la competencia que no tiene MecanoUOC, y puede realizar un preestudio en el que aparecería lo que necesita MecanoUOC para conseguir un producto igual.

Esquema conceptual ATD

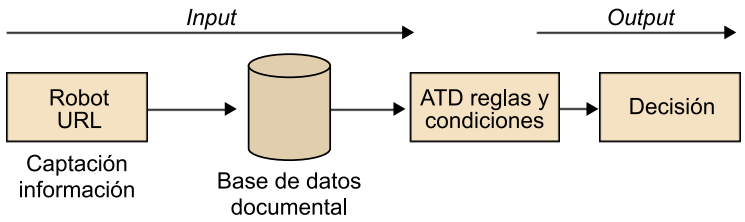


Figura 14

3.7. Informática

Todas las aplicaciones y servicios informáticos de MecanoUOC están centralizados en el Departamento de Informática.

Los servidores heterogéneos, con sistemas operativos de red diferentes como Novell, Unix, Linux y Windows, están conectados a una red local Gigabit Ethernet conmutada por un *switch*.

La red local de MecanoUOC tiene dos zonas diferenciadas, debido a los niveles de seguridad informática requeridos. Por un lado, tiene una red interna en la que se encuentran los servidores y servicios de uso exclusivo interno y, por otro lado, tiene una red denominada DMZ (*De-militarized Zone*, es decir, 'zona desmilitarizada') donde se encuentran los servidores y servicios informáticos centrales que requieren un acceso muy específico y puntual de acceso a Internet.

Las dos redes o zonas están conectadas a un sistema de seguridad denominado *cortafuego* (*firewall*), que monitoriza todo el tráfico de datos de salida y de entrada y actúa mediante unas reglas introducidas y un sistema de aprendizaje.

Red MecanoUOC

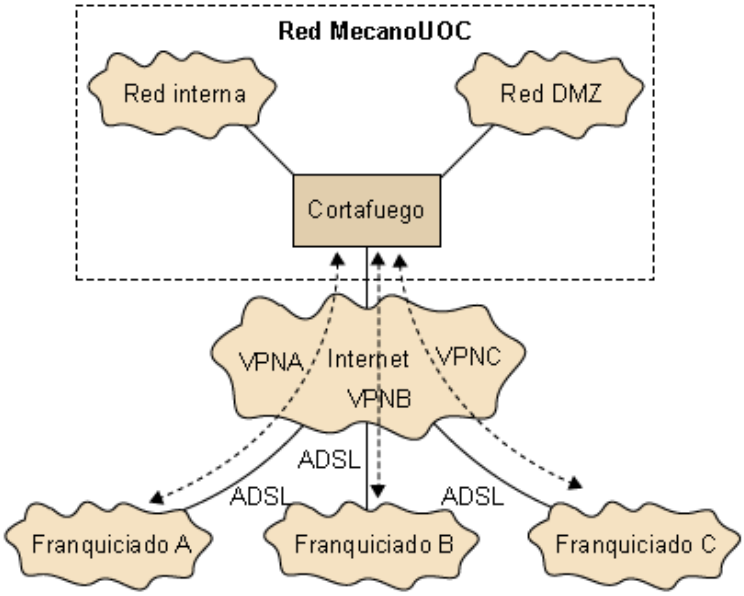


Figura 15

Cuando alguien quiere acceder desde Internet a los sistemas informáticos de MecanoUOC, el cortafuego identifica la solicitud de acceso y revisa si en las reglas introducidas en su base de datos propia se encuentra permitido este acceso, y a partir de aquí permite o no permite establecer la conexión.

Por otro lado, este cortafuego no actúa sólo permitiendo circular determinado tráfico de entrada o salida, sino que además tiene un sistema de aprendizaje que permite anticiparse y actuar ante un posible ataque. El cortafuego analiza el patrón, forma de conexión y tráfico de datos y determina si se trata de una conexión normal o anormal.

En caso de que determine que es una conexión anormal, podrá establecer un mecanismo de monitorización, el envío de alertas al administrador de la red y/o el bloqueo de la comunicación.

MecanoUOC dispone de una línea de acceso a Internet, y los franquiciados tienen líneas ADSL. El intercambio de datos entre MecanoUOC y los franquiciados se realiza de manera encriptada para garantizar la privacidad de los datos y su no repudio utilizando redes VPN (*Virtual Private Network*, es decir, 'redes privadas virtuales').

Además, el Departamento de Informática dispone de una serie de infraestructuras para ofrecer servicios informáticos a los trabajadores de MecanoUOC. Por ejemplo, dispone de un servidor de ficheros, de un sistema antivirus, de un

sistema de copias de seguridad para las aplicaciones y datos de los servidores centrales, de un servidor central de correo electrónico y de un servidor web, entre otras infraestructuras.